Giochi

Progetti hardware con lo ZX SPECTRUM

G. Bishop

Progetti hardware con lo ZX SPECTRUM

Progetti hardware con lo ZX SPECTRUM

G. Bishop

McGRAW-HILL Book Company GmbH

Amburgo · New York · St Louis · San Francisco · Auckland · Bogotá · Città del Guatemala · Città del Messico · Johannesburg · Lisbona · Londra · Madrid · Montreal · Nuova Delhi · Panama · Parigi · San Juan · San Paolo · Singapore · Sydney · Tokyo · Toronto

Ogni cura è stata posta nella creazione, realizzazione, verifica e documentazione dei programmi e dei progetti contenuti in questo libro. Tuttavia né gli Autori né la McGraw-Hill Book Co. possono assumersi alcuna responsabilità derivante dall'implementazione dei programmi stessi, né possono fornire alcuna garanzia sulle prestazioni o sui risultati ottenibili dal loro uso, né possono essere ritenuti responsabili di danni o benefici risultanti dall'utilizzo dei programmi. Lo stesso dicasi per ogni persona o società coinvolta nella creazione, nella produzione e nella distribuzione di questo libro.

Titolo originale: *Spectrum Interfacing and Projects*
Copyright © 1983 McGraw-Hill Book Co.(UK)Ltd - Maidenhead

Copyright © 1984 McGraw-Hill Book Co. GmbH - Hamburg

I diritti di traduzione, di riproduzione, di memorizzazione elettronica e di adattamento totale e parziale con qualsiasi mezzo (compresi i microfilm e le copie fotostatiche), sono riservati per tutti i paesi.

Realizzazione editoriale: EDIGEO snc, via Ozanam 10a, 20129 Milano
Traduzione: Giuseppe Zappalà
Grafica di copertina: Valentina Boffa
Composizione e stampa: Litovelox, Trento

ISBN 88-7700-005-8

1ª edizione Settembre 1984

ZX Spectrum e Microdrive sono marchi registrati della *Sinclair Research Ltd.*

Indice

Prefazione *7*

Uso dei programmi *9*

Capitolo 1 L'hardware dello Spectrum *11*
1.1 Lo schema del circuito *11*
1.2 Il cablaggio *13*
1.3 Lo schermo TV *17*
1.4 Il connettore *21*

Capitolo 2 Il software dello Spectrum *23*
2.1 Codice-macchina o BASIC? *23*
2.2 Il codice-macchina per i principianti *24*
2.3 Alcuni utili programmi in codice-macchina *29*

Capitolo 3 Conversione analogico-digitale *37*
3.1 Il comando IN *37*
3.2 Un convertitore analogico-digitale *38*
3.3 Particolari costruttivi dell'ADC *41*
3.4 Collaudo dell'ADC *44*
3.5 I joystick *48*
3.6 Un misuratore di luce *49*
3.7 Un sistema di esposizione fotografica *52*
3.8 Una penna ottica *54*
3.9 Un oscilloscopio digitale a memoria per lo spettro vocale *61*
3.10 Un termometro *64*

Capitolo 4 Conversione digitale-analogica *67*
4.1 La scheda latch *67*
4.2 Costruzione della scheda latch *70*
4.3 Collaudo della scheda latch *74*
4.4 Convertitore digitale-analogico *74*
4.5 Costruzione del DAC *77*
4.6 Collaudo del DAC *77*
4.7 Luci per l'albero di Natale o per la discoteca *80*
4.8 La subroutine "timer" *84*
4.9 Mini consolle per effetti di luce in teatro *86*
4.10 Sistema di riscaldamento domestico *87*
4.11 Generatore di forme d'onda *88*

Capitolo 5 Progetti ADC-DAC *91*
5.1 Progetti di registrazione e riproduzione del suono *91*
5.2 Lo Spectrum usato come registratore *93*
5.3 Lo Spectrum parlante *96*
5.4 Esperimenti di robotica controllati dallo Spectrum *97*
5.5 Una camera d'eco *110*
5.6 Sistema di controllo per trenini elettrici *113*
5.7 Gestione dell'energia domestica *117*
5.8 Sistema d'allarme antifurto *121*

Capitolo 6 Altri progetti *125*
6.1 Come resuscitare uno Spectrum bloccato *125*
6.2 Un amplificatore *126*
6.3 Una tastiera migliore *131*
6.4 Lo Spectrum in automobile *133*
6.5 Un alimentatore d'emergenza per lo Spectrum *135*

Appendice A I codici-macchina dello Z80 *137*

Appendice B Tabelle di conversione esadecimali-decimali *145*

Appendice C I segnali dello Z80 *149*

Appendice D Pin del connettore a pettine per i tre progetti principali *151*

Appendice E Dati sui componenti *153*

Appendice F Elenco dei componenti *155*

Appendice G Adattamenti per lo Spectrum 16K *157*

Appendice H Basette a circuito stampato *161*

Prefazione

Questo è un libro di computer diverso da qualunque altro. Non contiene giochi, invasori spaziali, missili o castelli incantati, ma permette a ogni possessore di uno Spectrum Sinclair di usare il computer a casa, in ufficio, per divertimento, come sussidio per gli handicappati, e anche di raggiungere una piena comprensione del funzionamento della macchina.
I progetti elettronici sono alla portata di un hobbista medio, costano poco ed hanno applicazioni praticamente illimitate.
I progetti si inseriscono nel connettore posteriore dello Spectrum senza danneggiarlo in alcun modo. Grazie a queste realizzazioni è possibile interfacciare lo Spectrum al mondo esterno, con applicazioni in fotografia, ricerca, istruzione, industria, affari, controllo dei processi, robotica ...
Alla base di questi progetti stanno due basette principali: un convertitore analogico-digitale (ADC) e un convertitore digitale-analogico (DAC), che contiene un circuito latch. Un progetto sfrutta il solo latch per accendere o spegnere fino a otto dispositivi, che possono essere tanto piccole lampadine quanto potenti motori elettrici. Il convertitore analogico-digitale legge di continuo otto segnali di ingresso, che possono essere sfruttati individualmente o nel loro insieme. La basetta principale contiene gli amplificatori necessari per questi circuiti.
Ovviamente nessuno di questi circuiti potrà funzionare senza l'opportuno software. Ogni progetto è, pertanto, accompagnato da un programma di gestione. È disponibile anche, separatamente, una cassetta contenente tutti i programmi. Molti suggerimenti di modifiche e aggiunte ai programmi sono disseminati lungo il libro.
Preliminarmente al lavoro di progetto, due capitoli introducono il lettore all'hardware dello Spectrum e alla programmazione in codice-macchina.

Il codice-macchina viene preferito al BASIC quando è necessaria velocità di elaborazione. Non lasciatevi intimidire dal codice-macchina; con la pratica scoprirete quanto sia facile (per chi sappia come e perché).
Durante la stesura di questo libro è stato fatto uso intensivo di dati sullo Z80, che sono stati inseriti come appendici al libro. Queste appendici saranno preziosissime al programmatore evoluto e a chi desideri studiare altri progetti hardware.
Anche se questo libro è stato scritto principalmente per lo Spectrum 48K, che mette a disposizione circa 40K di memoria, gran parte dei progetti potrà funzionare nella versione a 16K; le necessarie modifiche alle routine sono riportate nell'appendice G.
Sarà necessario caricare in macchina ogni programma, operare le necessarie modifiche, e quindi registrarlo di nuovo su nastro per la conservazione permanente. La connessione della stampante o dei Microdrive potrebbe essere impedita dalle basette collegate, ma si può evitare questo inconveniente sfruttando una basetta dotata di più connettori a pettine messi in parallelo. Prestate comunque la massima attenzione a non sovraccaricare l'alimentatore dello Spectrum.
Desidero ringraziare la Analogue Devices Ltd per l'uso dei dati del 7581, Colin Street e John Watson per la loro preziosa consulenza, Morris Bown per le fotografie, e specialmente mia moglie Julie e mia figlia Stella per l'aiuto fornitomi alla macchina da scrivere; desidero inoltre ringraziare tutta la famiglia e gli amici per la pazienza dimostratami durante la stesura del libro... la casa era sottosopra, egemonizzata dal computer, dalla macchina da scrivere e dal saldatore.

GRAHAM BISHOP

Uso dei programmi

Nel libro sono riportati i listati dei programmi: molti di essi contengono routine in codice-macchina caricate automaticamente nel programma BASIC; non è quindi necessario un LOAD CODE. Una volta caricato, il codice-macchina resterà immagazzinato fino alla cancellazione automatica conseguente al caricamento di un altro programma, o allo spegnimento del computer. Una routine di questo tipo è il programma "espanso" che deve essere caricato prima degli altri programmi (vedi per maggiori dettagli i capitoli successivi). Un programma in codice-macchina è in genere riconoscibile da linee di programma del tipo ... DATA 17,0,88... ed è spesso utile controllare se il codice sia stato effettivamente inserito nelle corrette locazioni prima di dare il RUN per non correre il rischio di perdere tutto.
Alcune linee di programma come:

```
1000 FOR x=0 TO 57
1001 PRINT PEEK (40000+x)
1002 NEXT x
```

possono essere aggiunte a qualsiasi programma, eseguite con un GOTO 1000 e successivamente cancellate. Ovviamente, il numero di byte (x) e l'indirizzo iniziale (40000) dovranno essere adattati al programma da controllare.
Ogni errore in una routine in codice-macchina causerà probabilmente il blocco del sistema.
I programmi sono stati scritti da un ingegnere elettronico: il sottoscritto.

Essi sono abbastanza strutturati, ma forse non verranno considerati belli o poetici dagli esperti di software.
La tecnica di programmazione adottata, comunque, permette aggiunte e modifiche con minimo sforzo; il testo stesso contiene, dove è necessario, opportuni consigli in merito.

Capitolo

L'hardware dello Spectrum

1

La Sinclair descrive così le caratteristiche dello Spectrum:

16K di BASIC (ROM)
16 o 48K di RAM
Capacità di gestire 8 colori
Generatore di suoni
Tastiera completa
Grafica ad alta risoluzione (256×176)
32 colonne per 24 righe di testo

Il manuale dello Spectrum fornisce inoltre le specifiche d'uso del connettore a pettine situato sul retro dell'apparecchio che rende accessibili le 16 linee di indirizzo, le 8 linee dati, le tensioni di alimentazione e alcuni segnali dello Z80 (vedi Fig. 1.2). Scopo di questo capitolo è chiarire il significato delle caratteristiche sopra esposte e dei segnali presenti sul connettore, la loro importanza ai fini delle operazioni interne dello Spectrum, e le loro possibilità d'uso nella connessione al computer di circuiti esterni.

1.1 Lo schema del circuito

La basetta del circuito stampato è molto complessa, dovendo interconnettere, su due facce, qualcosa come 26 circuiti integrati e vari altri compo-

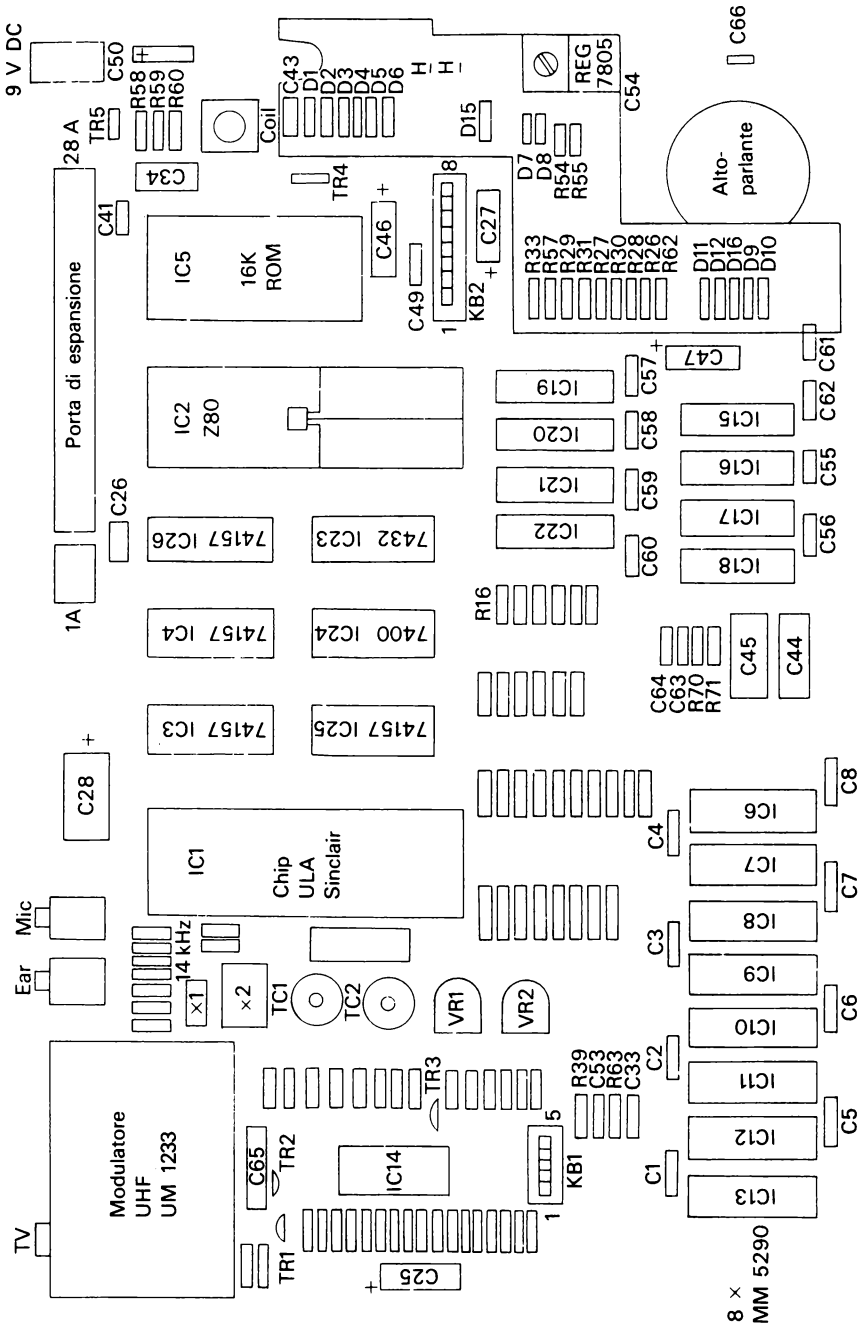


Figura 1.1 La basetta a circuito stampato dello Spectrum.

nenti. La disposizione dei circuiti sulla basetta è evidenziata in figura 1.1. Possiamo riconoscere le seguenti sezioni:

Il microprocessore Z80: il "cervello" del computer che è in grado, pilotato da un clock a 3.5 MHz, di eseguire fino a 900000 operazioni in codice-macchina al secondo.

16K di memoria a sola lettura (ROM) che contengono lo speciale BASIC dello Spectrum. Non è possibile eseguire dei POKE in questa memoria.

16K (o 48K) di memoria a lettura/scrittura (RAM). Si tratta di 8 circuiti integrati (16 nello Spectrum 48K) che costituiscono la memoria di lavoro del computer. Tre di questi chip vengono usati come memoria video, mentre i rimanenti 8K (o 40K) sono a disposizione dell'utente.

L'unità logico-aritmetica (ULA) Sinclair: uno speciale integrato che gestisce la memoria dello schermo e degli attributi.

Il modulatore UHF: il circuito che converte il segnale di riga generato dal computer in un segnale UHF idoneo a essere inviato nella presa d'antenna di un qualunque televisore.

7805: il regolatore a tre piedini che alimenta gran parte dei circuiti. È interessante notare che il ronzio che si sente quando lo Spectrum è acceso viene generato da un inverter da 5 a 12 V che è situato vicino al connettore dell'alimentazione a 9V CC.

KB1 e KB2: i due connettori della tastiera (vedi Cap. 6).

Lo Spectrum è uno strumento molto delicato che non dovrebbe mai essere manomesso. Pertanto non dovranno mai essere effettuate saldature o comunque collegamenti diretti al connettore a pettine. Per evitare (o, almeno, minimizzare) cortocircuiti e danni irreparabili, ogni collegamento andrà fatto tramite un connettore a 54 pin e basette a circuito stampato come quelle descritte nei successivi capitoli.

1.2 Il cablaggio

All'interno del computer corrono otto linee parallele di dati (D0-D7) lungo le quali vengono trasmessi gli 1 e gli 0. Su queste linee sarà sempre presente una combinazione di 0 e di 1 che rappresenterà un dato di valore compreso fra 0 e 255 in forma decimale; la rappresentazione binaria del numero 255 è 11111111. Queste otto linee sono sempre connesse ai vari dispositivi del computer. Sarà possibile osservare dei segnali ad alta frequenza, connettendo (con la massima attenzione) un oscilloscopio ad una delle linee D0-D7.

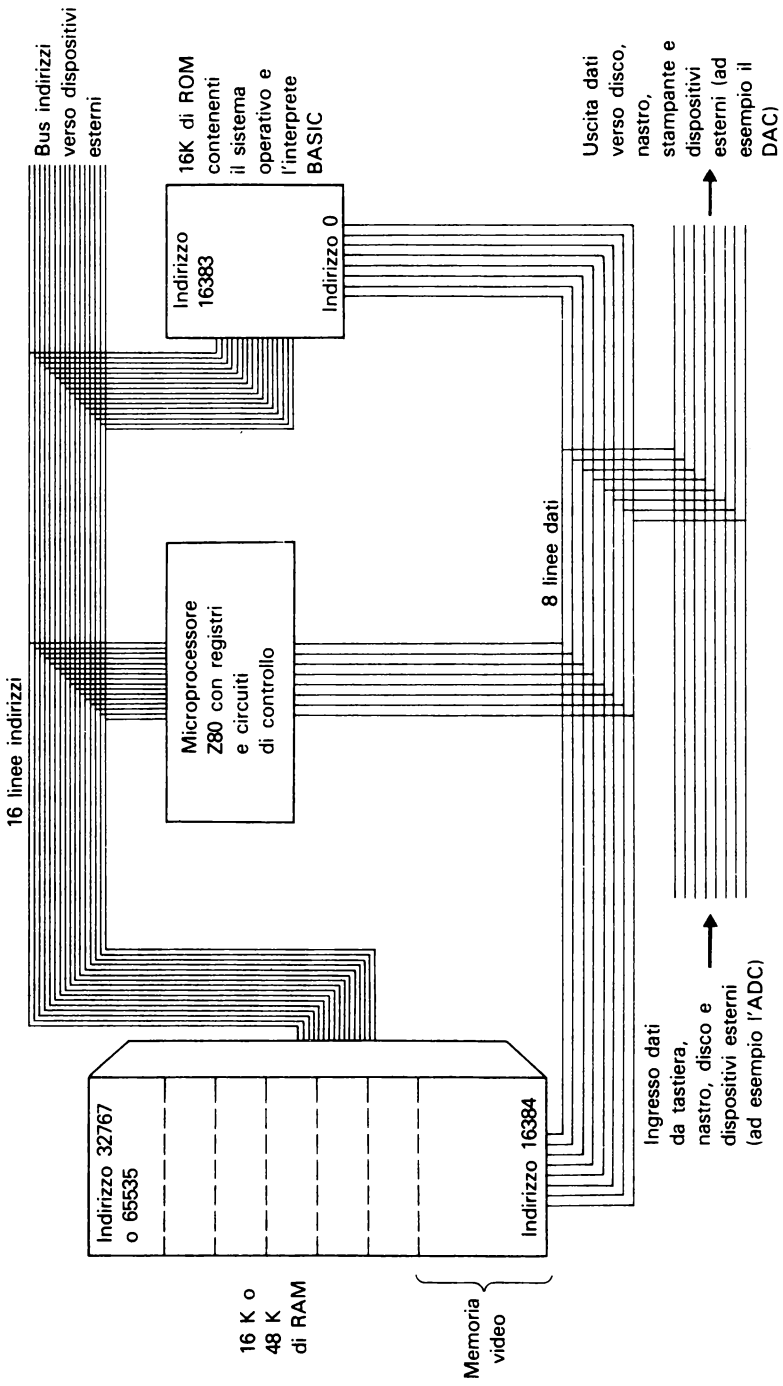


Figura 1.2 Le 8 linee dati e le 16 linee indirizzi dello Spectrum.

Per ottenere la corretta distribuzione di questi segnali, ogni dispositivo ed ogni locazione di memoria ha un indirizzo, simile a un codice di avviamento postale. Ogni dispositivo riconosce il suo proprio indirizzo, e, quando abilitato, trasmette gli otto bit di dati sul bus dati (WRITE), riceve gli otto bit di dati presenti sul bus dati (READ), o azzera tutti i bit (RESET, CLEAR o NEW).
Nella figura 1.2 possiamo vedere le interconnessioni fra i componenti principali: RAM, ROM e Z80. Lo Z80 è il vero cervello del computer, che controlla ogni operazione per mezzo di sequenze predeterminate di operazioni in codice-macchina, immagazzinate dentro lo Z80. Per esempio, per copiare un byte (8 bit) da una locazione di memoria in un'altra, sono necessarie circa 20 operazioni estremamente elementari per aprire e chiudere le "porte" elettroniche presenti nel circuito. Queste operazioni vengono scandite molto velocemente, ed ecco spiegato l'uso del clock a 3.5 MHz.
La RAM (da 16K o 48K) è dotata di indirizzi organizzati come in figura 1.3, con semplici flip-flop in grado di contenere molti 0 ed 1. È possibile leggerne o variarne il contenuto, sfruttando un interruttore dati in grado di indirizzare ogni riga di flip-flop. Lo Z80 fornisce opportuni segnali di controllo per comunicare all'interruttore dati (abilitato e disabilitato da una opportuna decodifica di indirizzo) se leggere, scrivere o resettare (READ, WRITE, RESET). Quando la decodifica riconosce il proprio indirizzo, abilita l'interruttore dati, che verifica lo stato delle linee di controllo ed esegue l'operazione richiesta.
Nello Spectrum, le otto linee dati e le sedici linee indirizzi raggiungono, senza alcuna interfaccia, il mondo esterno attraverso il connettore a pettine posto sul retro del computer e andranno trattate con la massima cura. Lo Z80 possiede 24 registri, sfruttati tanto dallo Spectrum che dall'utente. I registri sono raggruppati (vedi Fig. 1.4) in coppie di registri da 8 bit (come la coppia A-F), o in registri da 16 bit come IX, IY e SP.

A chiamato anche accumulatore
F chiamato anche registro dei flag
H chiamato registro alto (high)
L chiamato registro basso (low)
DE chiamato coppia di registri destinazione
BC è generalmente usato per comandi di stampa
IR viene usato per indicizzazioni (come IX e IY)
SP viene talvolta chiamato puntatore di catasta (stack pointer) e contiene l'indirizzo della successiva locazione di memoria che sarà usata (con comandi di PUSH o di POP).

Attraverso il comando EXX sono disponibili otto registri alternativi, che aumentano lo spazio di lavoro. Le coppie di registri possono venire sfrut-

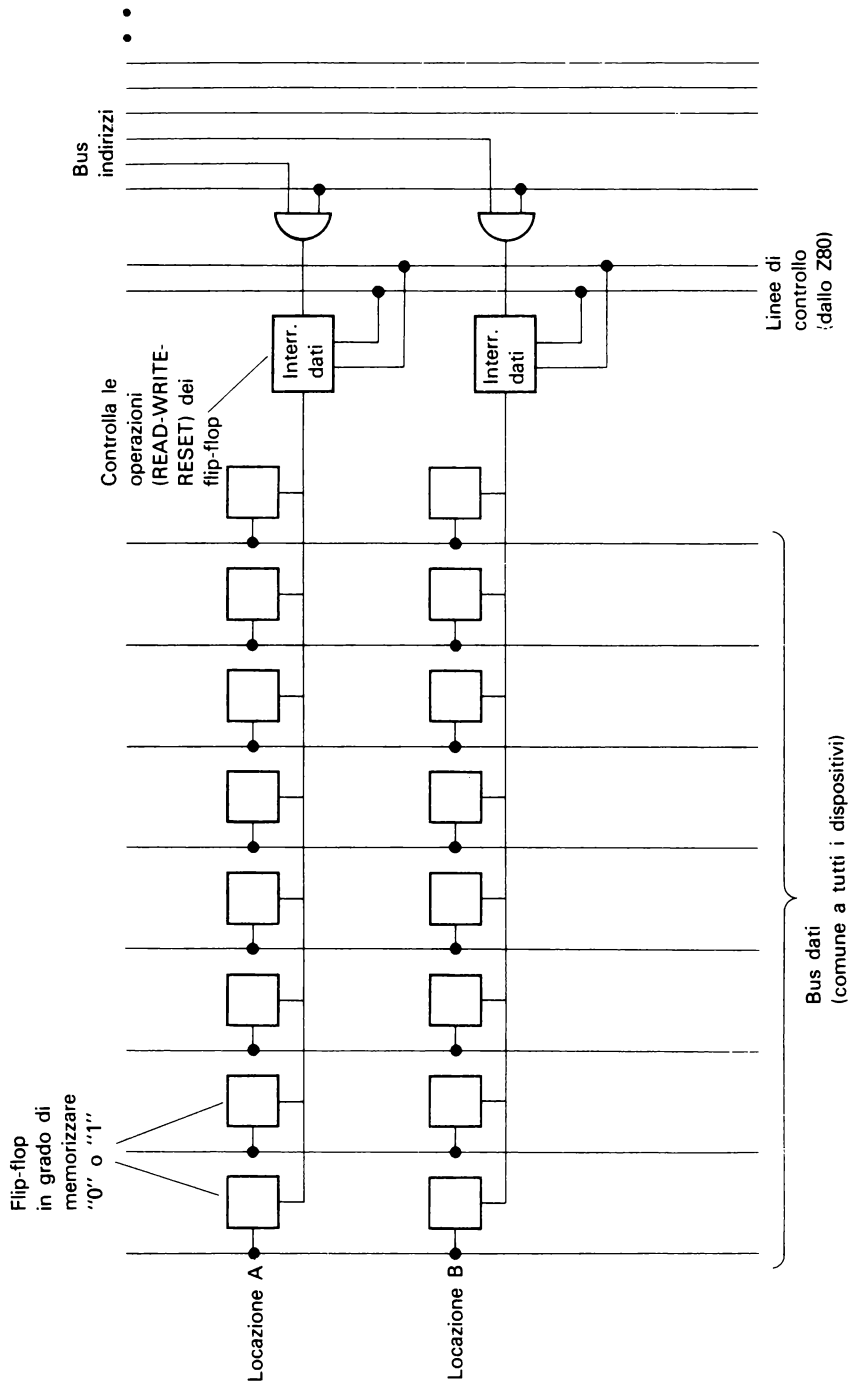


Figura 1.3 Lettura e scrittura di dati nelle locazioni di memoria.

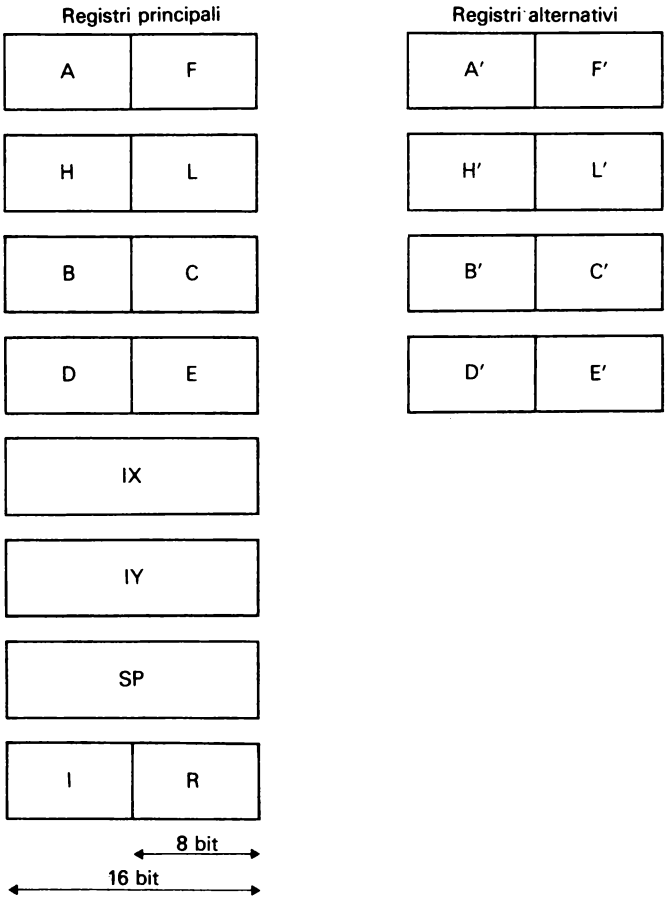


Figura 1.4 I 24 registri dello Spectrum.

tate come singoli registri da 8 bit o come registri da 16 bit. La struttura dei registri è analoga a quella di figura 1.3, con in più la possibilità di manipolarne il contenuto bit a bit, di ruotarlo verso destra o verso sinistra, di complementarlo (scambiando gli 1 e gli 0).

1.3 Lo schermo TV

Le locazioni di memoria comprese fra 16384 e 22527 sono riservate alla visualizzazione TV ad alta risoluzione come mappa in memoria.
L'utente può leggere dalla memoria video e scrivervi, indirizzando tanto

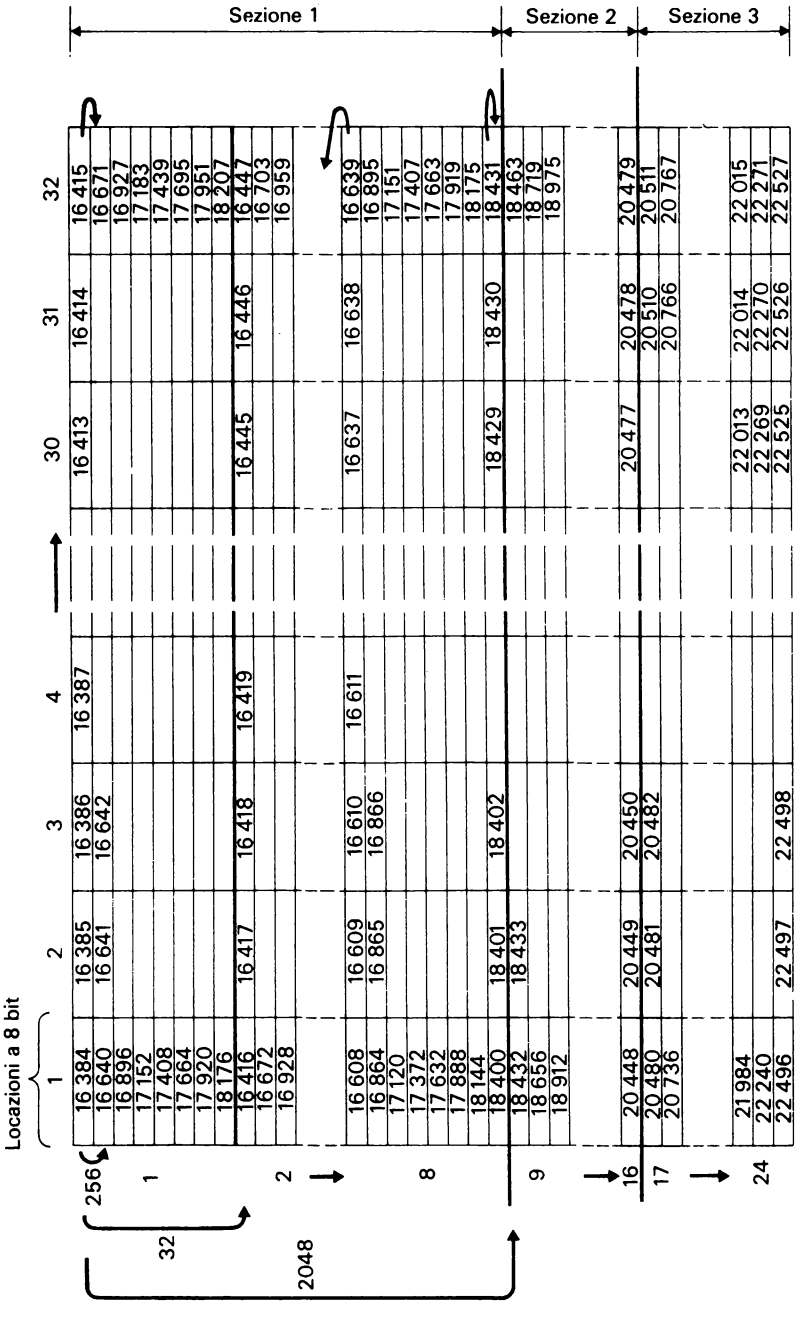


Figura 1.5 La memoria video dello Spectrum.

un singolo pixel o un gruppo di pixel per formare un carattere. I byte della memoria video non sono purtroppo organizzati secondo un semplice criterio matematico analogo alla scansione televisiva standard. In figura 1.5 vediamo la sequenza delle varie locazioni di memoria corrispondenti alle varie zone dello schermo.
Lo schermo è diviso in 32 colonne, numerate da 1 a 32. Ogni byte occupa l'ampiezza di una colonna, e contiene otto bit. Si hanno così a disposizione $8 \times 32 = 256$ pixel per riga.
Quando un bit è 1, il corrispondente pixel viene "acceso", quando è 0 il pixel resta "spento".
Digitate:

```
CLS:POKE 16384,128
```

e si accenderà un pixel nell'angolo superiore sinistro dello schermo. Il colore del pixel dipenderà dal valore di INK scelto. Digitando:

```
CLS:POKE 16415,1
```

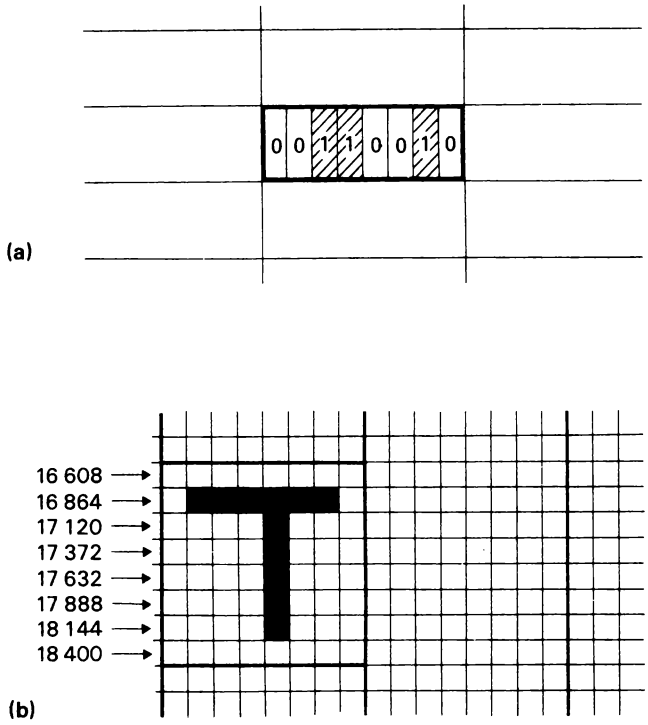
si accenderà l'ultimo pixel della prima riga. Digitando

```
CLS:POKE 16384,85
```

si accenderanno, alternati, quattro dei primi otto pixel della prima riga (85 viene scritto in binario come 01010101).
Per la scansione verticale, la sequenza è:

indirizzo iniziale 1ª riga: 16384
indirizzo iniziale 9ª riga: 16416
indirizzo iniziale 17ª riga: 16448
fino a
indirizzo iniziale 57ª riga: 16608, quindi
indirizzo iniziale 2ª riga: 16640
indirizzo iniziale 10ª riga: 16672, e così via.

Questa disposizione può sembrare molto complessa ad esempio quando si desidera ottenere un oggetto in rapido movimento sullo schermo! Anche se questo libro non è dedicato a giochi, diremo che il video mappato in memoria mostra la sua utilità quando, ad esempio, si desidera salvare in memoria un intero schermo usando il codice-macchina come spiegato nel capitolo 2. Nella figura 1.6a possiamo vedere un byte da otto bit, e nella figura 1.6b gli otto byte necessari per formare la lettera "T". Il manuale Spectrum descrive il procedimento da seguire per definire simboli grafici per mezzo della funzione BIN.




```
POKE   16 608, 0
       16 864, 126
       17 120, 8
       17 372, 8
       17 632, 8
       17 888, 8
       18 144, 8
       18 400, 0
```

Figura 1.6 (a) Un byte della memoria. (b) Le locazioni di memoria che compongono la lettera "T".

Fino ad ora abbiamo trattato solo della visualizzazione in nero su fondo giallo chiaro, con cornice neutra. I colori e gli effetti vengono aggiunti nell'area di memoria destinata agli attributi (le locazioni comprese fra 22528 e 23295). Come si vede in figura 1.5, ogni attributo controlla un blocco di otto byte; sono disponibili 32 blocchi orizzontali e 24 blocchi verticali. Non è possibile colorare un singolo pixel o un singolo byte. Tutti i 64 byte di un blocco devono usare lo stesso colore. La struttura interna dei byte di attributo è la seguente:

Bit	0, 1, 2	colore del carattere (ink), compreso fra 0 e 7 (in binario)
	3, 4, 5	colore dello sfondo (paper), compreso fra 0 e 7 (in binario)
	6	bright, 1 per "acceso", 0 per "spento"
	7	flash, 1 per "acceso", 0 per "spento"

Ad esempio:

```
CLS:POKE 22528,184
```

visualizzerà caratteri neri lampeggianti su fondo bianco (l'equivalente binario di 184 è 10111000).
Per vedere tutte le combinazioni possibili, digitate:

```
10 CLS:LET y=0
20 FOR x=22528 TO 23295
30 POKE x,y
40 LET y=y+1
50 IF y>255 THEN LET y=0
60 NEXT x
```

La variabile y assumerà tutti i valori da 0 a 255.

1.4 Il connettore

I segnali di uscita sul connettore a pettine sono generati da circuiti CMOS (Complementary Metal Oxide Semiconductor). L'impedenza di uscita di questi segnali è molto alta e rende quindi necessari circuiti di interfaccia per proteggere i componenti dello Spectrum. Il convertitore analogico-digitale (ADC) e il convertitore digitale-analogico (DAC) provvedono a questa necessità.
Sul connettore posteriore sono disponibili alcune tensioni di alimentazione (−5V, 9V, 12V e −12V), in grado, purtroppo, di fornire pochissima corrente; se vengono usate, lo Spectrum tende ad andare in blocco. L'unico contatto da cui si possa prelevare un poco di corrente è a 5V; anche questa andrà usata con estrema cautela, non alimentando mai più di due

o tre integrati. L'eccessivo carico, oltre a causare malfunzionamenti, potrebbe danneggiare permanentemente il regolatore. In caso di necessità (come nei progetti presentati nel capitolo 4) di correnti maggiori di quelle erogabili dallo Spectrum, si dovranno sfruttare alimentatori esterni.

Capitolo

Il software dello Spectrum

Lo Spectrum può essere programmato tanto in BASIC che in codice-macchina Z80; normalmente si usano entrambi i tipi di programmazione. I programmi in codice-macchina muovono i dati direttamente fra i vari registri dello Z80. Il programma riceve segnali e dati dalla tastiera e dalle porte di ingresso e genera segnali di uscita verso lo schermo televisivo o le porte di uscita. I programmi in codice-macchina lavorano molto velocemente, eseguendo 875000 operazioni al secondo. Per esempio, le circa 6000 parole da 8 bit della memoria video, equivalenti a 48000 pixel possono essere spostate da una zona di memoria all'altra e di nuovo in quella originale in una frazione di secondo. I videogiochi fanno uso intensivo di routine in codice-macchina per spostare le navi spaziali, le palle da tennis o altri oggetti in lungo e in largo per il terreno di gioco.

2.1 Codice-macchina o BASIC?

Ci sono 3 livelli di programmazione per lo Spectrum: il BASIC, l'Assembler e il codice-macchina. L'Assembler usa un assemblatore per interpretare i mnemonici (elencati nell'appendice A) e fa evitare di battere stringhe di numeri apparentemente senza senso, in istruzioni DATA; ogni numero corrisponde ad un'operazione in codice-macchina.
Molti utenti dello Spectrum preferiscono scrivere i loro programmi esclusivamente in BASIC, più lento del linguaggio macchina, nel quale l'esecuzione di ogni comando richiama una dopo l'altra varie routine in

codice-macchina. Ad esempio, in un semplice programma BASIC per controllare un pixel del video:

```
10 PLOT 50,100
```

le routine in codice-macchina per ogni "parola" potrebbero essere le seguenti:

10	La ROM del BASIC converte i numeri di linea in indirizzi di memoria lunghi due byte che vengono immagazzinati in ordine progressivo così da far eseguire poi in sequenza le varie operazioni richieste.
PLOT	La ROM del BASIC chiama un programma monitor, comprendente un certo numero di operazioni in codice-macchina sui registri, immagazzinati in memoria come parole da 8 bit e 16 bit.
50	Il numero viene convertito dall'istruzione PLOT del monitor in una coordinata x relativa al video mappato in memoria (vedi par. 2.2). Questo carattere viene riconosciuto dal monitor come fine del numero 50 ed inizio del numero 100. Provando, potete facilmente constatare che ogni altro carattere genera un messaggio di errore.
100	Il secondo numero viene convertito anch'esso da PLOT nella coordinata y del video mappato in memoria.
ENTER	Questo tasto identifica la fine del numero 100. Dato il RUN, il monitor forzerà ad 1 (binario) l'appropriata locazione di memoria (x,y), così da accendere il pixel selezionato dalla coppia 50,100 (vedi par. 1.3). Alla fine di questo programma, verificata l'assenza di ulteriori numeri di linea, il monitor mette il computer in stato di attesa (OK).

I circuiti di questo libro sono controllati da routine scritte tanto in BASIC che in codice-macchina, la cui comprensione è raccomandabile, mentre si ritiene scontata una valida conoscenza operativa del BASIC.

2.2 Il codice-macchina per i principianti

Non lasciatevi impressionare dai programmi in codice-macchina
Gli utenti soliti a programmare in BASIC lo troveranno probabilmente noioso ed oscuro. Il codice-macchina è essenziale dove è richiesta rapidità di esecuzione. Volendo infatti conservare in memoria un'intera immagine, è meglio che l'operazione sia eseguita in un trentesimo di secondo

che in trenta secondi. Il caricamento ed il salvataggio con SCREEN$ sono prova evidente di quanto asserito.
In sostanza, per programmare in codice-macchina, è necessario ragionare in termini estremamente elementari. Non esistono numeri di linea ed occorre usare una sequenza di locazioni di memoria, scelte a partire da circa 24000 fino alla RAMTOP, pari a 32767 per lo Spectrum 16K e a 65535 nella versione 48K. È consigliabile lasciare abbastanza spazio per il programma BASIC che inizia in 23760; l'introduzione di codice-macchina sul BASIC bloccherebbe il funzionamento del sistema. La locazione 32000 (7D00 esadecimale) per la versione 16K e una locazione compresa fra 64000 e 65280 (FA00-FF00 esadecimale) per la versione 48K, sono indirizzi da cui potrebbero iniziare le routine in codice-macchina.
Caricando programmi in codice-macchina è consigliabile lasciare un po' di spazio fra i vari programmi ed effettuare un preliminare CLEAR per ripulire la memoria al di là dell'indirizzo di partenza.
Per esempio, CLEAR 32000 ripulirà tutte le locazioni di memoria successive alla 32000 compresa, e ne inibirà l'uso da parte del BASIC.
Esistono vari metodi per caricare il codice-macchina:

1. Usando un *loader* esadecimale come:

```
10 LET d=32000
20 DEF FN a (a$,b)=CODE a$(b)-48 -7*(CODE a$(b)>57)
30 DEF FN c(a$)=16*FN a(a$, 1)
   +FN a(a$,2)
40 READ a$:IF a$< >"**" THEN
   POKE d, FN c(a$):LET d=d+1: GO TO 40
50 DATA "00", "01", "0F", .........."**"
```

   La linea 50 contiene il codice-macchina in esadecimale che, convertito in decimale, viene immagazzinato a partire dalla locazione 32000.
2. Usando istruzioni REM per riservare spazio per il codice-macchina all'inizio del programma BASIC:

```
1 REM ..... 50 caratteri qualsiasi.....
10 DATA 6, 192, 17, 10, 0, 33,..... (50)
20 FOR x=23760 TO 23810
30 READ v:POKE x,v
40 NEXT x
```

   Questo esempio tratta 50 dati (linee 10 e 20) e va introdotto nella macchina prima del caricamento del programma principale, che chiamerà la routine in codice-macchina tramite:

```
LET l=USR 23760
```

Le linee 10-40 possono essere cancellate prima di caricare il programma principale, lasciando il codice-macchina nello spazio di memoria occupato dalla REM di linea 1.
Caricato il codice-macchina, nel listare il programma noteremo una strana serie di caratteri in linea 1: si tratta della rappresentazione sullo schermo del codice-macchina appena caricato.
Importante è l'indirizzo di partenza 23760, che dovrebbe essere sempre usato. Un vantaggio di questo metodo è che il codice-macchina è immagazzinato al sicuro insieme al programma BASIC ed è così preservato da danneggiamenti sempre possibili in altre aree di memoria.

3. Caricando il codice-macchina durante il caricamento del programma BASIC, in modo da effettuare una sola operazione:

```
 10 FOR x=32000 to 32050
 20 READ m:POKE x,m
 30 NEXT x
 40 DATA 1, 99, 0, 33......
100 programma BASIC
 .
 .
 .
```

Il numero di elementi presenti nel DATA deve ovviamente corrispondere al numero di locazioni di memoria allocate (51 in questo esempio), altrimenti verrà visualizzato il messaggio OUT OF DATA ERROR.
Digitando RUN, verrà caricato il codice-macchina, e quindi avrà inizio l'esecuzione del programma BASIC, che userà

```
LET l=USR 32000
```

per chiamare la routine in codice-macchina.
Dei vari metodi preferisco il terzo perché:

1. È semplice in quanto consiste di sole tre linee.
2. Non usa, come il primo metodo, la notazione esadecimale facendo risparmiare così una laboriosa digitazione e riducendo le occasioni di errori matematici frequenti nei meno esperti.
3. È di veloce caricamento (circa un secondo) e non impone di fermare l'esecuzione del programma principale per cancellare linee di programma.

Per chiamare, dall'ambiente BASIC, la routine in linguaggio macchina, è possibile, oltre alla LET l=USR 32000, usare l'istruzione

```
RANDOMIZE USR 32000
```

Sarà necessaria grande cura nella digitazione poiché un blocco del sistema generato dal codice-macchina non è recuperabile; conviene quindi salvare sempre i programmi su nastro prima di dare il RUN.
Una routine in codice-macchina caricata con il terzo metodo resta in memoria anche dopo il caricamento di un'altra routine o di un altro programma BASIC (purché ciò avvenga in altra zona di memoria, ovviamente). Un programma BASIC può allora essere salvato su nastro in modo usuale, tramite

SAVE "Nome" oppure con
SAVE "Nome" LINE 1

se si desidera la sua esecuzione automatica appena caricato o tramite

SAVE "Nome" CODE

se è presente in memoria solo un programma in codice-macchina o tramite

SAVE "Nome" CODE indirizzo iniziale, lunghezza

se si vuole salvare solo una parte della memoria
La programmazione in codice-macchina comprende:

1. Caricamento dei registri HL, BC, DE con le costanti.
2. Conteggio, confronto e movimento di dati incrementando, decrementando, sommando, ecc.
3. Ingresso o uscita di dati ai o dai registri, in particolare l'accumulatore. Molti programmi immagazzineranno dati in locazioni della memoria video, come ad esempio nella 16384, la cella iniziale di visualizzazione.
4. Chiamate delle routine del monitor dello Spectrum, ad esempio, CALL-DRAW.
5. Un comando di return, senza il quale il sistema si bloccherebbe, o, peggio, impazzirebbe. Il RET restituisce al BASIC il controllo dal programma in codice-macchina.

Nell'appendice A è presentato il set completo di istruzioni in codice-macchina Z80 dello Spectrum. Esso comprende i codici decimali (usati per il caricamento attraverso istruzioni DATA), i codici esadecimali (se si preferisce usare un loader esadecimale), e il numero di byte (che sarà 1 in caso di semplici movimenti di dati, 2 se è richiesto un numero N o X da un byte, 3 se è richiesto un numero NN da due byte). Insieme ai mnemonici è presentata una breve descrizione delle operazioni svolte.

Spesso si usa la coppia di registri HL per puntare ad una locazione di memoria o per alterarne il contenuto; ad esempio,

```
ld(hl),NN
```

caricherà il numero esadecimale NN nell'indirizzo corrispondente al numero contenuto nella coppia di registri HL. La coppia di istruzioni

```
ld hl,00FF
ld(EE00),hl
```

forzerà il valore 255 (00FF in esadecimale) nella coppia di registri HL, col byte di ordine alto a 0 e il byte di ordine basso contenente 255; quindi la locazione 60928 (EE00) riceverà da HL il numero 255 e la locazione 60929 riceverà il numero 0.

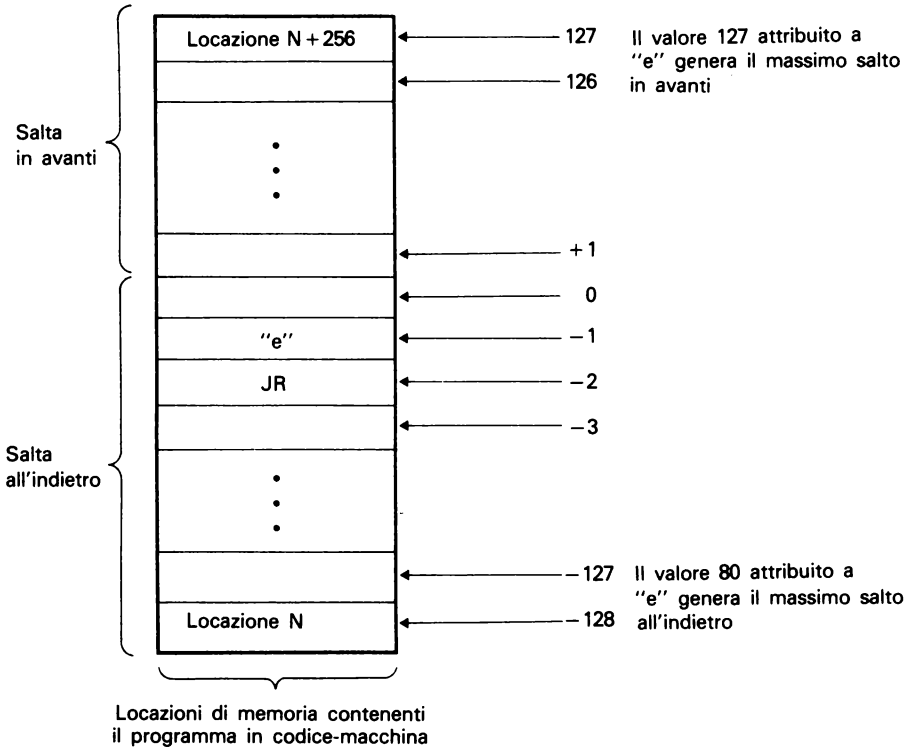


Figura 2.1 La procedura di conteggio per le istruzioni di salto relativo in avanti e all'indietro.

Scrivendo i numeri NN, il byte di ordine basso va digitato prima di quello di ordine alto; l'esempio appena presentato apparirà quindi come:

	Esadecimale	*Decimale*
ld hl,00FF	21	33
	FF	255
	00	0
ld(EE00),hl	22	34
	00	0
	EE	238

Il computo dei byte, quando si desidera incrementare o diminuire il contenuto del registro contatore di programma — usando ad esempio l'istruzione jr nz, x — è alquanto complesso (vedi Fig. 2.1). I salti in avanti o all'indietro implicano sempre il computo dei byte oppure — usando jp NN — il salto ad una particolare locazione.

2.3 Alcuni utili programmi in codice-macchina

I progetti contenuti in questo volume richiedono l'uso di un certo numero di programmi in codice-macchina, di cui vi diamo i listati e le spiegazioni.

COME COPIARE DAL VIDEO ALLA MEMORIA

Codifica	*Decimale*	*Risultato*
ld de,58327	17	indirizzo di memoria
(E3D7 in esadec.)	215	=
	227	58327
ld hl,16384	33	indirizzo base della memoria video
(4000 in esadec.)	0	=
	64	16384
ld b,27	6	inizializzazione del contatore
	27	B(1)...
push bc	197	...e suo salvataggio

ld b,0	6 0	inizializzazione del contatore B(2) a 256 (=0)
ld a,(hl) ld(de),a inc de inc hl	126 18 19 35	sposta, byte a byte, i dati dalla memoria video al buffer
djnz−7	16 249	decrementa B(2) e salta 7 locazioni indietro se B(2) non è zero
pop nc	193	riprende B(1)
djnz−12	16 244	decrementa B(1) e salta 7 locazioni indietro se B(1) non è zero
ret	201	ritorna al video

I contatori nel programma arrivano a 27 e a 256. Quando 27×256=6912 locazioni sono state spostate, il programma termina. Ciò sposta tutti i 6144 byte dello schermo mappato in memoria, assieme ai 768 byte degli attributi che contengono le informazioni sui colori.

COME COPIARE NELLA MEMORIA VIDEO IL CONTENUTO DI UNA ZONA DI MEMORIA

Codifica	*Decimale*	*Risultato*
ld de,16384 (4000 in esadec.)	17 0 64	indirizzo base della memoria video =16384
ld hl,58327 (E3D7 in esadec.)	33 215 227	indirizzo base del buffer di memoria =58327
ld b,27	6 27	inizializzazione del contatore B(1)...
push bc	197	...e suo salvataggio
ld b,0	6 0	inizializzazione del contatore B(2) con 256
ld a,(hl) ld(de),a inc de inc hl	126 18 19 35	copia i dati dal buffer alla memoria video

djnz – 7	16 249	decrementa B(2) e salta se B(2) non è zero
pop bc	193	riprende B(1)...
djnz – 12	16 244	lo decrementa e salta se B(1) non è zero
ret	201	ritorna al BASIC

Da notare il doppio uso che viene fatto in questi due programmi del registro B, come B(1) e B(2), salvandone (PUSH) il contenuto sullo stack e riprendendolo (POP) in seguito per tornare al contatore B(1).
Con questo metodo è possibile immagazzinare, in banchi di 7K, varie immagini in una frazione di secondo. Comunque, lo Spectrum 16K può immagazzinarne solo una, ad esempio, dalla locazione 25088 in poi, col codice-macchina che inizia in 32000, lasciando ben poco spazio libero. La versione 48K può agevolmente immagazzinare quattro immagini di questo tipo.

QUANTA MEMORIA RESTA DISPONIBILE?

Questo programma vi comunica quanti byte di memoria avete ancora a disposizione per i vostri programmi.

Codifica	*Decimale*	*Risultato*
ld hl,0000	33 0 0	azzera HL
add hl,sp	57	salva lo stack pointer in HL
ld de,(23653) (5C65 in esadec.)	237 91 101 92	carica in DE l'indirizzo della locazione di memoria che contiene il puntatore all'inizio della RAM libera = 23653
and a	167	
sbc hl,de	237 82	HL ← HL – (DE) calcola lo spazio libero
push hl pop bc	229 193	passa il risultato in BC (via stack)
ret	201	ritorna al BASIC

La routine in codice-macchina appena presentata può essere caricata in memoria per mezzo del seguente programmino e chiamata con

```
PRINT USR 32000
10 FOR x=0 TO 13
20 READ n
30 POKE (32000+x), n
40 NEXT x
50 DATA 33, 0, 0, 57, 237, 91, 101, 92, 167, 237, 82, 229,
   193, 201
```

Potete, se preferite, sfruttare una REM per contenere il codice-macchina:

```
1  REM aaaaaaaaaaaaaa
10 DATA 33, 0, 0, 57, 237, 91, 101, 92, 167, 237, 82, 229,
   193, 201
20 FOR x=23760 TO 23773
30 READ n:POKE x,n
40 NEXT x
```

In questo caso, la chiamata della routine in codice-macchina è:

```
PRINT USR 23760
```

Indipendentemente dal metodo scelto, la routine resta chiamabile in qualunque istante durante l'esecuzione dei programmi.

PROGRAMMA "ESPANSO"

Presenteremo altri programmi via via che si renderanno necessari al funzionamento dei nostri progetti elettronici. Presentiamo ora una routine che trova impiego in molti progetti. Essa permette la visualizzazione di caratteri (lettere, numeri o simboli di qualunque misura) in qualunque posizione dello schermo.
I parametri sono mostrati in figura 2.2 e possono essere inseriti in un semplice programma BASIC:

```
10 LET xs=4:  LET ys=4:  LET cs=8:  LET y=100:
   LET d$="Spectrum":
   GOSUB 9390

9390 LET x=(256-xs* cs* LEN d$)/2
```

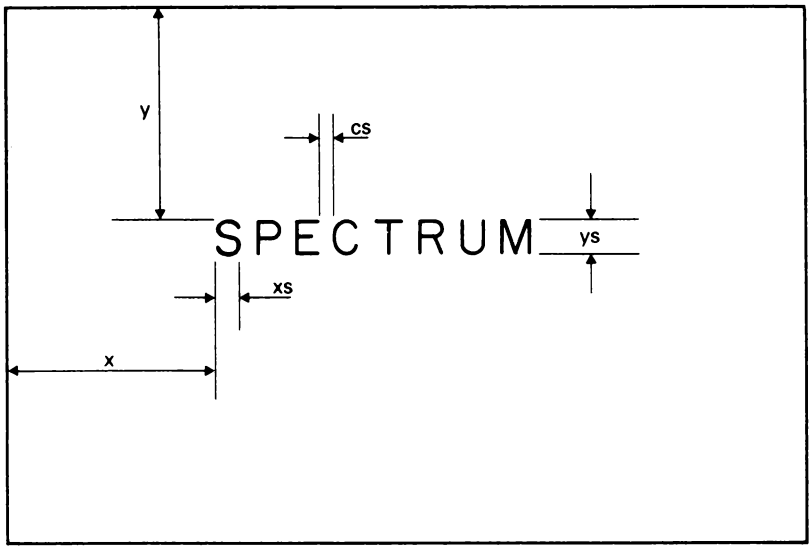


Figura 2.2 I parametri del programma "espanso":
y da 0 a 175
x da 0 a 255
xs, ys, cs possono assumere un qualsiasi valore (20 fa riempire lo schermo con un solo carattere)

9400 LET a=23306: POKE a,x: POKE a+1,y: POKE a+2,xs: POKE a+3,ys: POKE a+4,cs: LET a+4: FOR i=1 TO LEN d$: POKE a+i, CODE d$ (i): NEXT i: POKE a+i, 255: RANDOMIZE USR 32256; RETURN

Il corrispondente programma in codice-macchina può essere caricato nel modo seguente:

```
   1 CLEAR 45500
   2 FOR x=32256 TO 32532
   3 READ a: POKE x,a
   4 NEXT x
   5 DATA 33,15,91,126,35,34,0,91,111,60,200,
38,0,41,41,41,237,75,54,92,9,62,8,50,4,91,58,
11,91,50,9,91,58,10,91,50,8,91,62,9,50,5,91,1
26,35,34,2,91,7,50,6,91,58,5,91,61,32,50,58,4
```

(continua)

```
,91,61,32,24,58,14
   6 DATA 91,71,58,12,91,79,58,10,91,129,5,32
,252,50,10,91,42,0,91,195,3,126,50,4,91,58,13
,91,71,58,9,91,128,50,9,91,42,2,91,195,32,126
,50,5,91,58,12,91,71,58,9,91,50,7,91,58,13,91
,79,197,205,164,126,193,58,7
   7 DATA 91,60,50,7,91,13,32,241,58,8,91,60,
50,8,91,5,32,221,58,6,91,195,48,126,128,64,32
,16,8,4,2,1,58,142,92,238,255,71,58,141,92,16
0,71,58,8,91,230,248,111,58,7,91,254,192,208,
31,31,31,230,31,103,203,28,203,29,203,28,203,
29,203,28,203,29,62,88,180
   8 DATA 103,58,142,92,166,176,119,58,7,91,7
1,230,7,246,64,103,120,31,31,31,230,24,180,10
3,120,23,23,230,224,111,58,8,91,71,31,31,31,2
30,31,181,111,235,33,156,126,120,230,7,79,6,0
,9,70,26,33,6,91,203,70,40,3,176,18,201,47,17
6,47,18,201
  10 LET xs=4: LET ys=4: LET cs=8: LET y=100:
 LET d$="Spectrum": GO SUB 9390
  20 STOP
9390 LET x=(256-xs*cs*LEN d$)/2
9400 LET a=23306: POKE a,x: POKE a+1,y: POKE
a+2,xs: POKE a+3,ys: POKE a+4,cs: LET a=a+4:
FOR i=1 TO LEN d$: POKE a+i,CODE d$(i): NEXT
i: POKE a+i,255: RANDOMIZE USR 32256: RETURN
```

Questo programma può essere fuso al programma principale, o salvato su nastro come codice-macchina:

SAVE "espanso" CODE 32256,277

Per ricaricare il codice-macchina, digitate

LOAD "espanso" CODE 32256,277

e quindi il solito LOAD " " per caricare il programma BASIC.

È possibile ottenere effetti grafici molto interessanti cambiando i colori dei caratteri dello sfondo e della cornice e sfruttando il comando FLASH. Per usare questo programma con quelli presentati nei capitoli successivi dovrete seguire il seguente procedimento:

1. Caricate "espanso" in modo usuale, fermando il nastro ad operazione completata.
2. Premete RUN. Questo comando caricherà il codice-macchina nelle locazioni 32256-32532, dove resterà permanentemente immagazzinato. Sullo schermo apparirà la scritta "Spectrum" seguita dal comando STOP.
3. Caricate il programma principale (cioè quello che usa la nostra routine). Potreste ad esempio usare a tale scopo il programma "foto" presentato nel capitolo 3, oppure eseguire un MERGE con qualunque altro programma, caricando prima "espanso" e dando poi il comando di MERGE. Per esempio, col programma "tempo" sottoriportato è necessario solo caricare le linee 10-70, mentre le linee 9390 e 9400 possono rimanere dalla fusione col programma "espanso". Prestate la massima attenzione a non duplicare numeri di linee essenziali al funzionamento del secondo programma; una linea 10 nel secondo programma cancellerà la linea 10 della nostra routine. I numeri 9390 e 9400 sono stati scelti per ospitare la routine BASIC che lavora col codice-macchina di "espanso" ma sarebbe andata bene qualunque altra coppia: la nostra subroutine si trova all'inizio dei programmi, così da non interferire. La linea 9390 aggiusta automaticamente la posizione orizzontale dei caratteri, calcolando il valore di x. La linea 9400 provvede al resto. Se l'utente desidera imporre ad x un valore predeterminato, dovrà cancellare la linea 9390 o sostituirla con un'altra routine, come la linea 80 del programma "foto", dove x=20+(70*(z−1)), e sfruttare poi (eventualmente con un GOSUB) la linea 9400.

Un'applicazione di questa routine sfrutta le locazioni di temporizzazione 23674, 23673 e 23672 per ottenere un orologio digitale.

```
  10 POKE 23672,0: POKE 23673,0: POKE 23674,0
  11 LET m=0: LET s=0
  20 LET t=INT ((65536*PEEK 23674+256*PEEK 23
673+PEEK 23672)/50)
  30 IF s=59 THEN  CLS
  40 LET m=INT (t/60): LET s=t-(m*60)
  45 IF m=1 AND s=30 THEN  BEEP 1,10
  50 LET ys=4: LET xs=4: LET cs=8: LET y=10:
LET d$=STR$ m: GO SUB 9390
  60 LET y=100: LET d$=STR$ s: GO SUB 9390
  70 GO TO 20
```

(continua)

```
9390 LET x=(256-xs*cs*LEN d$)/2
9400 LET a=23306: POKE a,x: POKE a+1,y: POKE
a+2,xs: POKE a+3,ys: POKE a+4,cs: LET a=a+4:
FOR i=1 TO LEN d$: POKE a+i,CODE d$(i): NEXT
i: POKE a+i,255: RANDOMIZE USR 32256: RETURN
```

Questo programma sarà usato in seguito per applicazioni fotografiche in camera oscura.

Capitolo

Conversione analogico-digitale

3

Lo Spectrum, come ogni altro computer digitale, tratta solo segnali digitali. Dispositivi esterni, come la stampante ZX e i Microdrive, possono essere collegati allo Spectrum purché i segnali assumano la forma di una serie di parole composte da 8 bit. Questo capitolo descrive un circuito d'interfaccia fondamentale: il convertitore analogico-digitale (ADC) con alcune sue applicazioni.

3.1 Il comando IN

Lo Spectrum accetterà segnali provenienti dal mondo esterno in risposta al comando BASIC

IN indirizzo della porta

Per sfruttare questa caratteristica occorre applicare un segnale ad otto bit alle terminazioni dati del connettore a pettine dello Spectrum. Questi terminali sono connessi direttamente al bus dati ad 8 bit che interconnette i vari dispositivi interni del computer (ROM, RAM, CPU Z80...). Un modo di controllare l'attività interna dello Spectrum consiste nel collegare un oscilloscopio ad uno dei terminali D0-D7 del connettore a pettine. Non osserverete alcun segnale fin quando non manderete in esecuzione un programma, quindi potrete vedere segnali alla frequenza di vari MHz, la cui forma ha significato solo durante le operazioni di trasferimento di dati. Prestate la massima cura a non cortocircuitare fra loro o a massa le

linee D0-D7, pena alterazioni dei dati e, eventualmente, blocco del sistema.
Prestate sempre la massima attenzione quando manipolate il connettore a pettine o vi effettuate collegamenti. Contatti fra le linee di alimentazione e il bus dati o il bus indirizzi danneggeranno permanentemente il vostro computer.
Per introdurre dati nel computer occorre programmare la macchina affinché interrompa le operazioni in corso, cerchi i dati forniti, li accetti, e quindi riprenda quanto sospeso. È pertanto necessario un circuito di interfaccia. L'interfaccia ha un indirizzo che è scritto nel programma di gestione. Si può usare ogni indirizzo tra 0 e $2^{15}-1$, cioè 65535, semplicemente ponendo degli 0 o degli 1 sulle 16 linee di indirizzo A0-A15. Se, per esempio, vogliamo usare l'indirizzo 32, il computer dovrà generare il seguente segnale:

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1

L'interfaccia reagisce a questo segnale per mezzo di una semplice circuiteria e i dati vengono posti sul bus dati D0-D7. Tutte queste operazioni avvengono in circa un microsecondo, è quindi necessario usare circuiti integrati veloci, tipo TTL o CMOS.
Volendo, ad esempio, introdurre il numero 255 (11111111 binario) dalla porta 31, è necessario un circuito di interfaccia che, abilitato da un singolo impulso (0) su A5, ponga sul bus dati i nostri 1 binari. Un semplice programma BASIC di controllo potrebbe essere:

```
10 PRINT AT 0, 0; IN 31
20 GOTO 10
```

In questo esempio il computer leggerà di continuo il numero presente sulla porta di indirizzo 31 e scriverà 255 nell'angolo superiore sinistro dello schermo.

3.2 Un convertitore analogico-digitale

Molti segnali di ingresso al computer sono di tipo analogico, cioè variabili in un insieme continuo. A questa classe appartengono i segnali di uscita da sensori di temperatura, luce, suono e anche quelli provenienti da controlli a distanza dotati di alimentazione interna o esterna.
Il primo progetto descrive un convertitore analogico-digitale ad otto canali, dotato di otto ingressi separati e campionamento automatico dei lo-

ro valori in otto locazioni di memoria dello Spectrum. Il circuito è costruito intorno all'integrato 7581, che ha le seguenti caratteristiche:

1. Connessione diretta allo Spectrum (compatibilità hardware).
2. Alimentazione 5V prelevata dallo Spectrum.
3. Tempo di acquisizione di un ingresso pari a 50 μs, tempo di acquisizione per tutti gli otto ingressi di 400 μs.
4. Tensioni di ingresso ammissibili nel campo 0-10 V.
5. Ingresso diretto delle uscite digitali in otto indirizzi consecutivi dello Spectrum; in questo progetto sono usati gli indirizzi da 24 a 31.
6. RAM doppia porta 8×8 integrata nel chip.

Ulteriori particolari sul 7581 sono riportati in Appendice E. Lo schema elettrico del circuito è riportato in figura 3.1 e lo schema di montaggio in figura 3.2. Vengono usati tre circuiti integrati: un 7581, un 74LS27 ed un 40106. Il circuito è molto semplice. Il bus dati dello Spectrum è connesso direttamente al 7581 cui fanno capo anche le linee A0, A1 e A2 del bus indirizzi. La logica di decodificazione dell'indirizzo sfrutta due NOR gate del 74LS27, cui confluiscono, per essere opportunamente trattate, le linee A5, $\overline{RD}$ e $\overline{IORQ}$.
La "tabella della verità" è:

A5	$\overline{RD}$	$\overline{IORQ}$	CS	$\overline{CS}$
0	0	0	1	0
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

Il 7581 attende uno stato logico 0 sul CS (chip select: abilita/disabilita il chip). Quando ciò avviene, trasmette il risultato della conversione del canale otto nella porta 31. Come possiamo vedere dalla tabella della verità, affinché ciò avvenga è necessario che tanto $\overline{RD}$ (read: segnale attivo basso che comunica alla memoria che sta per essere letta) che $\overline{IORQ}$ (input/output request: quando questa linea è bassa, l'indirizzo presente sulle linee del bus indirizzi appartiene a una porta di ingresso/uscita N.d.T.) siano a zero logico. È possibile selezionare i canali 1-7 ponendo degli zeri sulle linee A0, A1, A2:

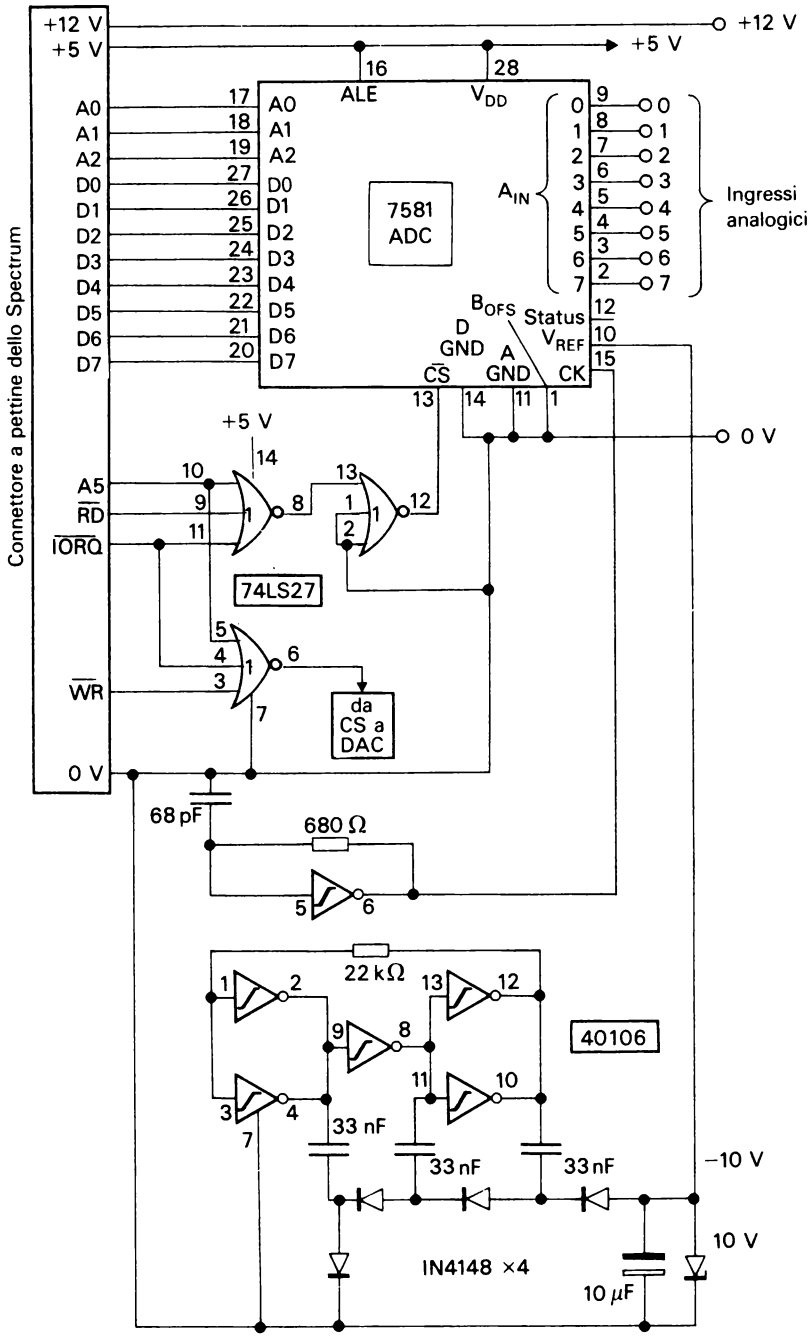


Figura 3.1 Lo schema elettrico del convertitore analogico-digitale.

	Indirizzo della porta di input	
Canale	*8*	31
	7	30
	6	29
	5	28
	4	27
	3	26
	2	25
	1	24

I valori (compresi tra 0 e 10 V) dell'ingresso analogico sono convertiti dal 7581 in valori discreti (numeri interi) compresi fra 0 e 255. Il 7581 ha il piedino 16 (ALE) connesso a 5 V, il piedino 10 (V_{REF}) connesso a -10 V, e il piedino 1 (B_{OFS}) connesso a 0 V. Questo tipo di funzionamento viene detto unipolare, essendo dotato di uscita binaria diretta.
È anche possibile ottenere uscite binarie bipolari oppure complementari, ma esse non sono state usate in questo progetto.
Il 7581 richiede un clock da 1.6 MHz sul piedino 15: a tale scopo viene sfruttato uno dei sei Schmitt inverter presenti nel 40106. I cinque inverter rimanenti vengono sfruttati, insieme ad una catena diodi-condensatori, per ottenere un moltiplicatore di tensione; vengono impiegati 3 condensatori da 33 nF e 4 diodi IN4148; il condensatore di smorzamento è da 10 μF e il diodo zener da 10 V.
La terza porta del 74LS27 viene usata come circuito di abilitazione della scrittura per il convertitore digitale-analogico che verrà presentato nel capitolo 4. Sulla basetta ADC è disponibile un connettore a 12 (o 16) ingressi per il collegamento diretto del convertitore digitale-analogico (DAC) allo Spectrum.

3.3 Particolari costruttivi dell'ADC

Come molti circuiti integrati CMOS, il 7581 è estremamente sensibile alle cariche statiche; occorrerà quindi prestare la massima attenzione nel manipolarlo. È pertanto opportuno che venga trasportato con i piedini "affogati" in schiuma conduttiva e che tutta l'attrezzatura sia adeguatamente messa a terra.
È disponibile un'idonea basetta a circuito stampato per il convertitore analogico-digitale (vedi App. H). In alternativa, è possibile montare il circuito su una basetta universale tipo RS 95.5 mm SRBP idonea all'impie-

go con integrati e componenti discreti, che sarà usata ulteriormente in altri progetti presentati in questo libro. In questo progetto non sfrutteremo il connettore placcato oro, che dovrà essere quindi accuratamente tagliato con un seghetto per evitare che lo Spectrum resti sollevato di circa un centimetro dal piano di appoggio. Questo accorgimento, pur essendo antiestetico, non danneggerà né la basetta né il computer.
La basetta viene collegata allo Spectrum per mezzo di un connettore doppia faccia a 28 poli, dotato di *chiave di polarizzazione* nella quinta sede. Il connettore usato è idoneo al *wire-wrap*; data la difficile reperibilità di questo tipo di connettore, potreste trovarvi costretti ad acquistare connettori più grandi (ad esempio 2×43 poli) da tagliare poi a misura con il seghetto, tenendo sempre presente la chiave, che dovrà tassativamente trovarsi in posizione cinque. Consigliamo di usare connettori da wire-wrap in quanto più adatti al collegamento con la basetta.
Per montare il circuito è innanzitutto necessario asportare dalla basetta il connettore a pettine placcato oro (per prezioso che possa sembrarvi, vale solo poche lire). Tranciate quindi le piste nei punti segnati ( ) in figura 3.2. Se non avete reperito in commercio il connettore 2×28 vie, rifilate opportunamente quello in vostro possesso, e tagliate alla base tutti i terminali non in uso.
Dato che i piedini tagliati non ricrescono, è opportuno prestare la massima attenzione a questa operazione necessaria ad evitare che i segnali non usati possano eventualmente generare problemi tanto allo Spectrum che alla basetta ADC. Provvedete ora a sistemare il connettore sul lato componenti della basetta e a saldarlo, lasciando sporgere i piedini di circa 3 mm dalla superficie ramata, in modo che siano solidamente fissati. La figura 3.3 potrà rivelarsi chiarificatrice. Saldate ora gli zoccoli per i circuiti stampati. Nel caso non trovaste lo zoccolo a 28 piedini necessario per il 7581 potrete ricavarlo tagliando uno zoccolo a 40 piedini, o in altri modi. Saldate poi su piste alterne le boccole da 2 mm attraverso cui saranno collegati gli otto ingressi analogici, la massa e l'alimentazione a 12 V.
Passiamo ora al cablaggio. Purtroppo i computer usano per le otto linee dati parallele e non, come ad esempio i sistemi audio, un solo filo. Occorrerà pertanto un gruppo di otto fili che interconnetta tutti i componenti. Inoltre, poiché la piedinatura dei circuiti integrati non organizza in sequenza i terminali dati e i terminali indirizzi, non sarà possibile usare le piste di rame per D0-D7 o A0-A15. Sarà allora necessario un complesso sistema di fili per collegare il connettore a pettine con i circuiti integrati, o il 7581 con i terminali di ingresso.
Il progetto minimizza il cablaggio, che comunque richiede 33 collegamenti con filo isolato in PVC e alcuni ponticelli in filo nudo, come è chiaramente visibile in figura 3.2. Infine, eseguito anche il cablaggio, si montano tutti gli altri componenti.

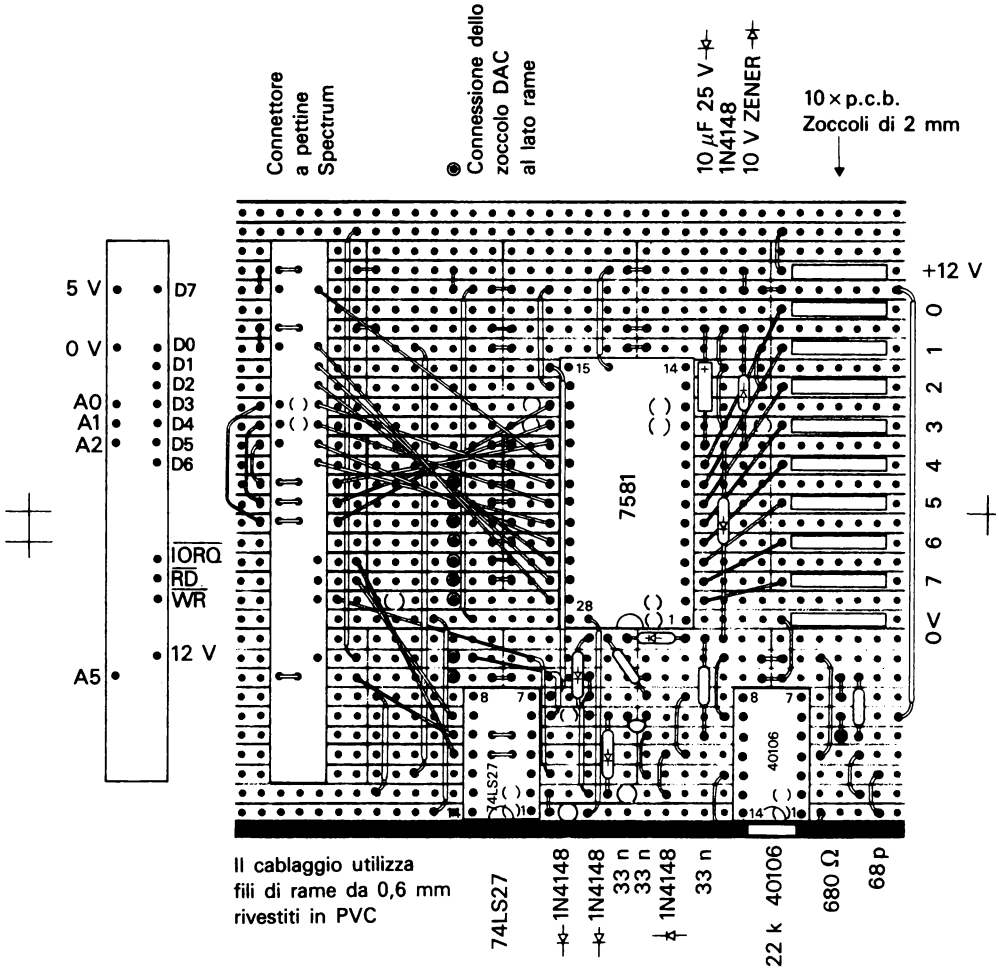


Figura 3.2 La realizzazione della basetta ADC.

Prima di inserire i circuiti integrati è opportuno stabilire se si adopererà il convertitore digitale-analogico presentato nel capitolo 4. In tal caso, è conveniente saldare sin d'ora un connettore a 12 o 16 vie sul lato rame della basetta (vedi Fig. 3.2 e Fig. 3.4). Come si vede anche in figura 3.5, il connettore per il DAC sarà sulla faccia opposta della basetta rispetto a quella su cui sono montati gli integrati e tutti gli altri connettori. È tuttavia possibile aggiungere in qualunque momento il connettore per il convertitore digitale-analogico.

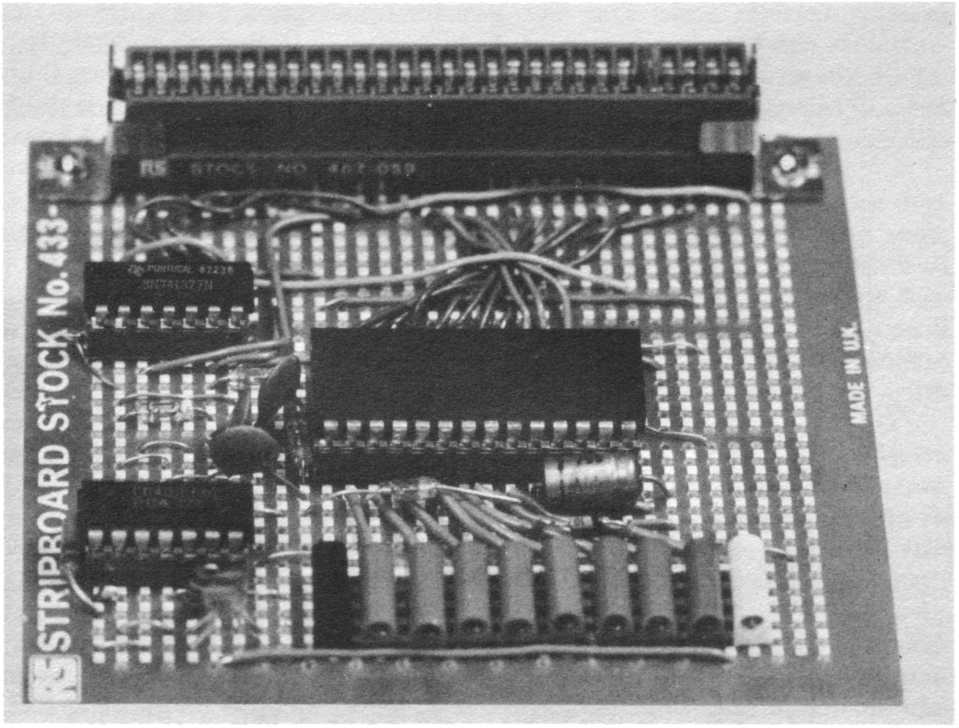


Figura 3.3 Basetta del convertitore analogico-digitale.

3.4 Collaudo dell'ADC

Il primo collaudo sarà visivo. Controllate attentamente di non avere commesso errori di cablaggio, cortocircuiti, saldature fredde, e altri errori ovvii. Mi è capitato di aver interpretato male il manuale dello Spectrum, cablando il connettore a pettine capovolto: fortunatamente il sistema non venne danneggiato. Comunque, cortocircuiti su alcune linee possono danneggiare permanentemente tanto il computer che il vostro montaggio.
Controllate ora, servendovi di un tester, l'assenza di cortocircuiti fra i terminali del connettore a pettine: una gocciolina di stagno può causare danni irreparabili.
Potete ora inserire la basetta nel connettore sul retro dello Spectrum e accendere il computer. Dovreste, anche a basetta inserita, poter seguire la solita procedura di accensione; in caso di esito negativo, sconnettete

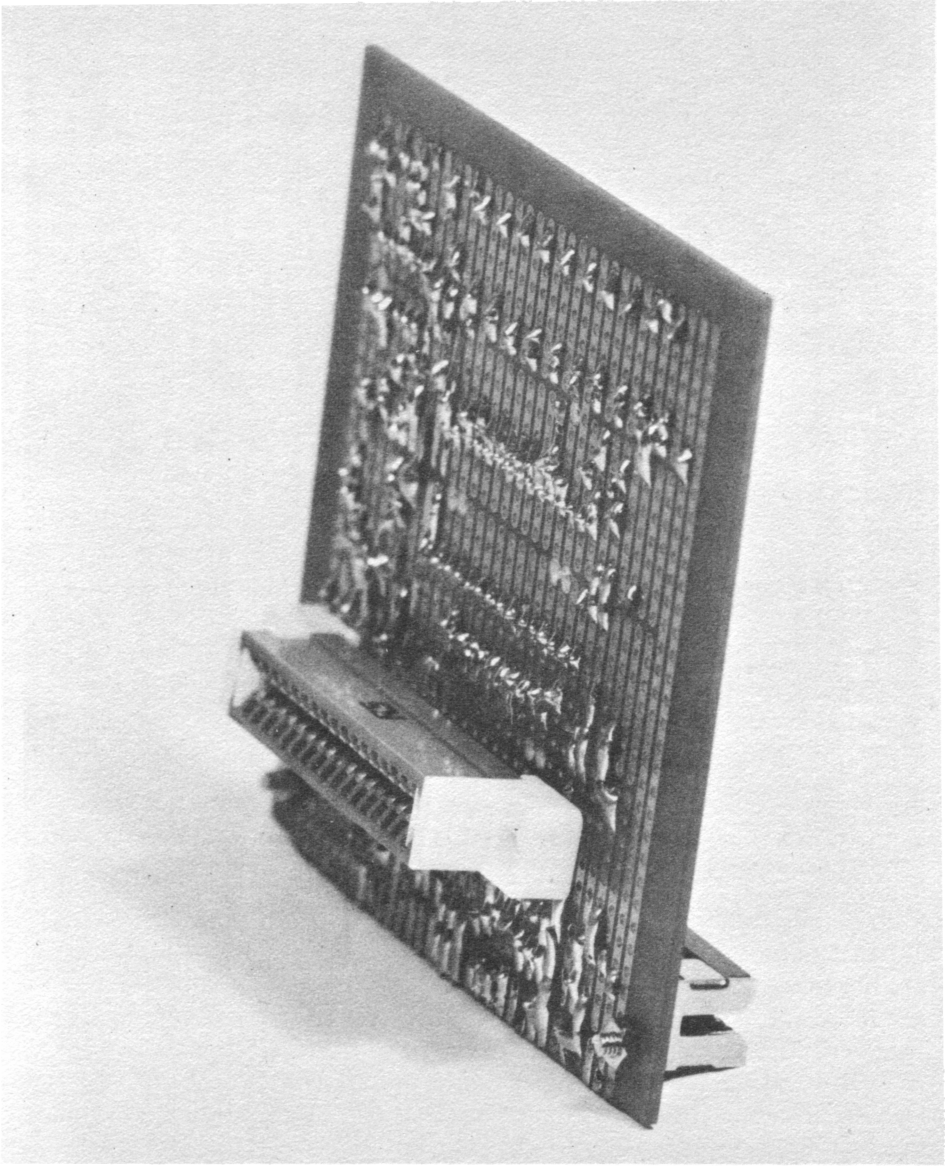


Figura 3.4 Il connettore a pettine sul retro della basetta ADC viene usato per collegare la basetta DAC.

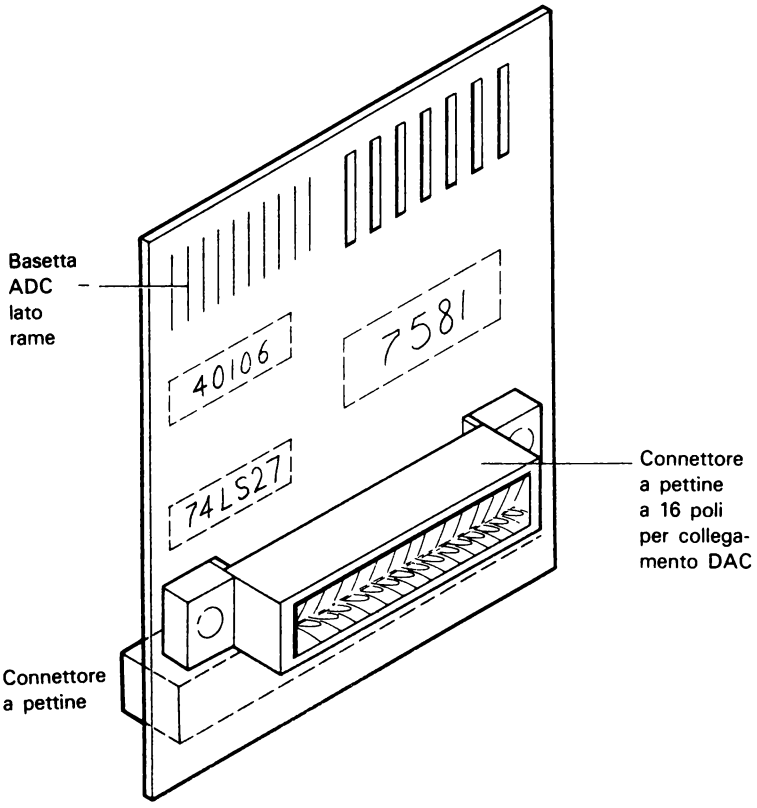


Figura 3.5 Il collegamento del connettore a sedici poli della basetta DAC sul retro della basetta ADC.

l'alimentazione e la basetta e ricontrollate il tutto molto attentamente. I segnali generati dall'ADC obbediscono alla seguente tabellina:

Canale	*Indirizzo della porta di input*
1	24
2	25
3	26
4	27
5	28
6	29
7	30
8	31

A convertitore scollegato, ogni indirizzo riporterà zero. Collegando l'ADC, senza fornirgli segnali in ingresso, ogni indirizzo riporterà 255. Ecco un semplice programma di prova:

```
10 CLS
20 FOR x=1 TO 10
30 PRINT AT x,0; (22+x), IN (22+x)
40 NEXT x
50 GOTO 20 (o GOTO 10 se desiderate ripulire ogni volta lo schermo).
```

Il risultato sullo schermo dovrebbe essere:

23	0
24	255
25	255
26	255
27	255
28	255
29	255
30	255
31	255
32	0

Prendete ora un filo e collegatene un estremo al connettore di massa. Inserendo di volta in volta l'altro estremo in uno dei connettori di ingresso dovreste produrre uno zero nella riga corrispondente dello schermo. (Data la particolare struttura della linea 30, vi apparirà 055, poiché ogni volta verrà cancellato solo il 2).
Se doveste riscontrare dei problemi di funzionamento, misurate le tensioni sulla basetta. Dovreste avere:

+ 5 V sul piedino 16 del 7581
+ 5 V sul piedino 14 del 74LS27
+ 5 V sul piedino 14 del 40106
− 10 V sul piedino 10 del 7581
+ 12 V sulla boccola di uscita.

Gli errori più comuni riscontrabili in questo stadio sono difetti di cablaggio, quindi continuate i controlli.
Il convertitore analogico-digitale è ora pronto per l'uso.

3.5 I joystick

Uno dei circuiti di più semplice costruzione fa uso di un joystick composto da due potenziometri da 100 kΩ fissati ad angolo retto fra di loro e da una cloche di comando (vedi Fig. 3.6).
Collegate il joystick all'ADC secondo lo schema di figura 3.7, curando di saldare i quattro connettori da 2 mm a quattro conduttori: 12 V, 0 V e a due qualsiasi degli ingressi, per esempio alle porte 24 e 25. Il programma di prova mostrerà variazioni comprese tra 0 e 255 corrispondentemente al movimento della levetta del joystick; desiderando "vedere" l'ingresso potrete usare il programma:

```
10 CLS
20 PLOT IN 24, (IN 25)*175/255
30 GOTO 20
```

Questo blocco può essere usato per ogni programma di giochi; la posizione sullo schermo è ottenuta direttamente da ogni coppia di indirizzi IN.

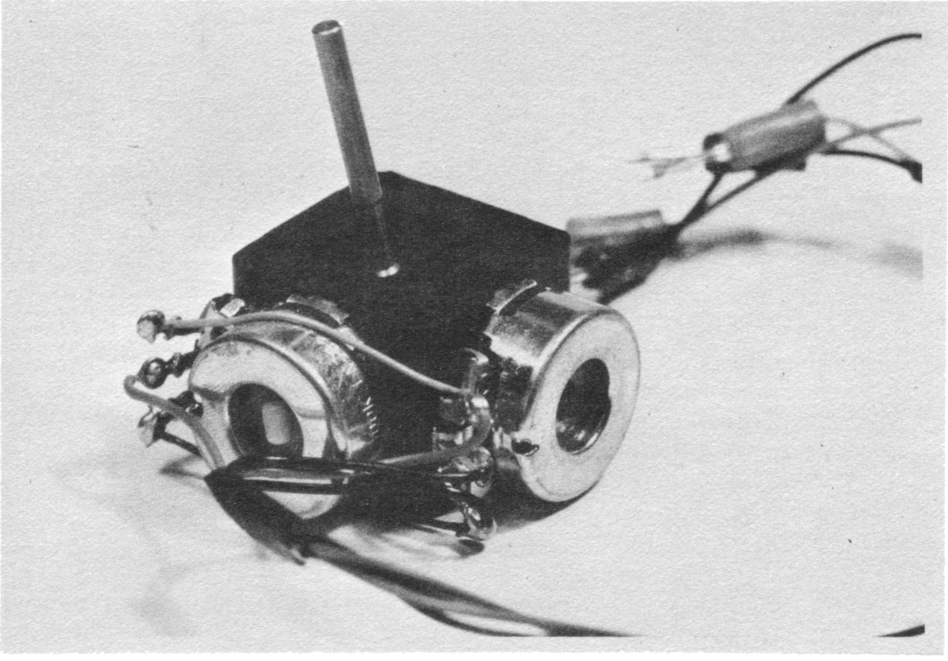


Figura 3.6 Joystick.

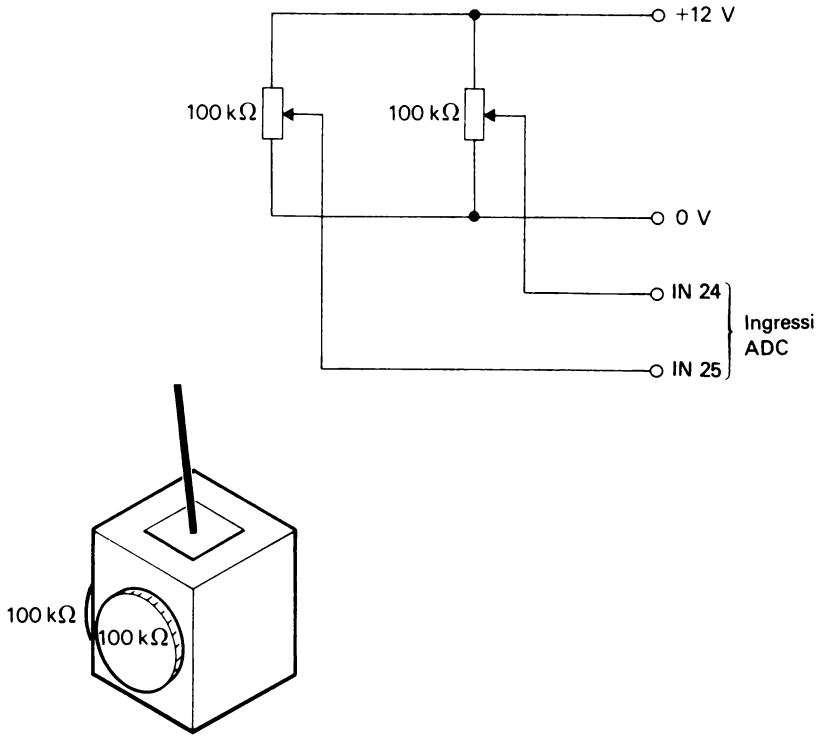


Figura 3.7 Connessione del joystick all'ADC.

3.6 Un misuratore di luce

Collegando un LDR (resistore variabile in funzione della luce) ad uno degli ingressi dell'ADC, otterrete, dopo opportuna calibrazione, un misuratore di luce (in lux), il cui responso spettrale potrà essere sfruttato in varie applicazioni. In figura 3.8 possiamo vedere il circuito formato da una resistenza da 6.8 kΩ e un LDR.
Un LDR è più robusto di una cella solare che richiederebbe inoltre un amplificatore per ottenere un adeguato interfacciamento con l'ingresso dell'ADC. La resistenza varia tra circa 20 Ω in condizioni di luce e 200 kΩ al buio; potrete quindi, con il circuito illustrato, ottenere variazioni di tensione fra circa 0V e 9V nel passaggio dalla luce al buio.
Desiderando ottenere letture in lumen o altra unità in misura della luce, dovrete usare opportune formule di conversione. Dovrete confrontare i

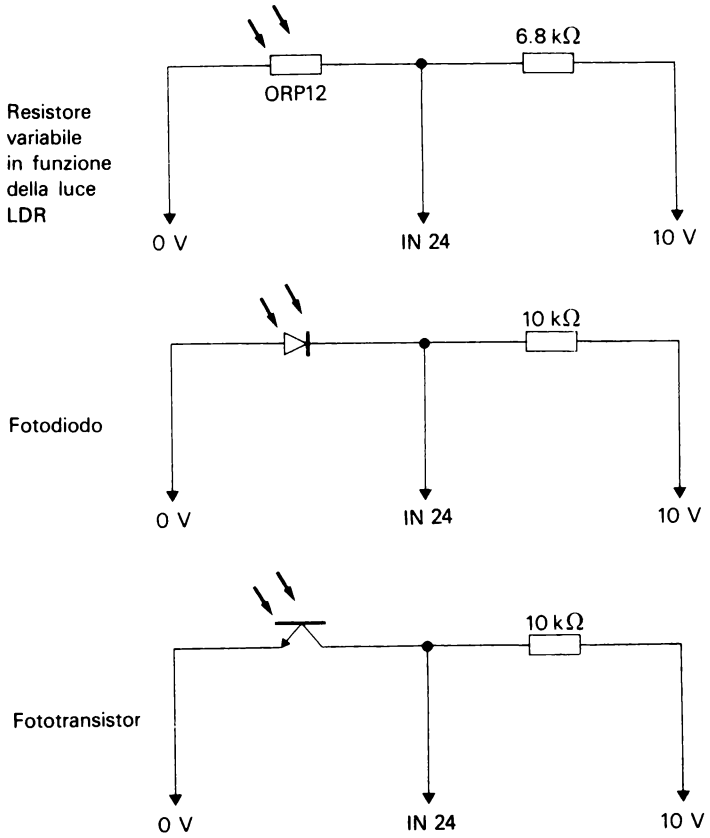


Figura 3.8 Uso di un LDR, fotodiodo o fototransistor per misurare livelli luminosi. Si può sfruttare qualunque ingresso della basetta ADC.

valori ottenuti con la lettura di un luxmetro e quindi inserire nel programma un'opportuna formula. Per esempio:

```
y=IN 24*x/255
```

Desiderando ottenere una visualizzazione in caratteri grandi potrete usare il programma "espanso" immettendo il codice-macchina nella memoria dello Spectrum e fondendo con "espanso" le seguenti linee di programma:

```
10 LET ys=4:  LET xs=4:  LET cs=8:  LET y=100:
   LET d$=STR$ IN 24: GOSUB 9390
20 GOTO 10.
```

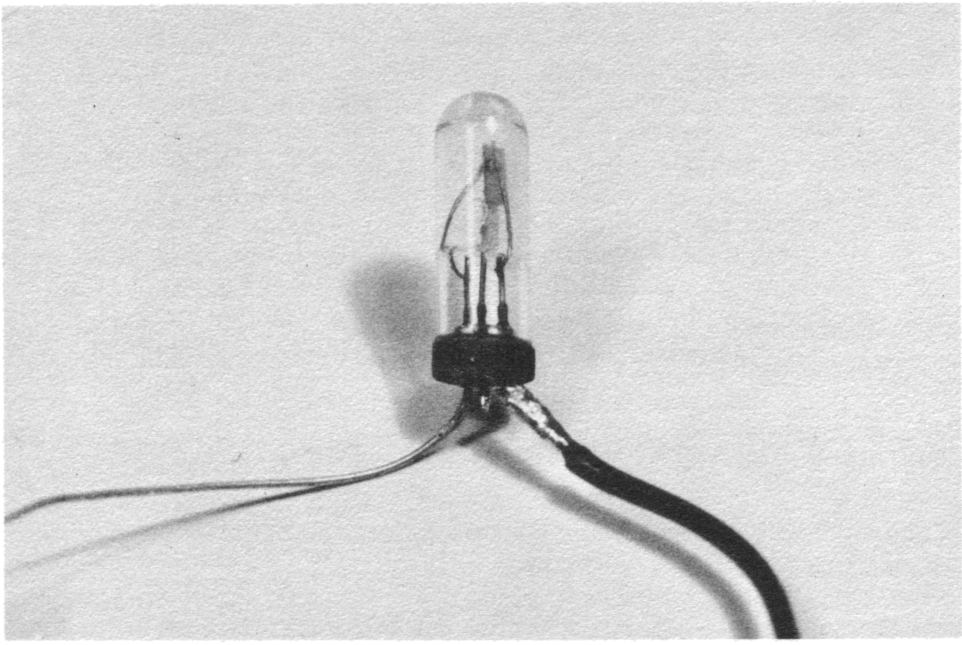


Figura 3.9 Tipico fototransistor per il misuratore di luce. Da notare la connessione della base in circuito aperto.

In alternativa, il valore calibrato di y può essere immesso in STR$.
È possibile ottenere un misuratore di luce anche servendosi di altri due dispositivi fotosensibili: un fotodiodo o un fototransistor. In figura 3.8 possiamo vedere le connessioni di questi dispositivi. Il fotodiodo è collegato con polarizzazione inversa fornita da un opportuno resistore in serie per produrre adeguate variazioni di valori sull'ingresso della basetta ADC. Il fototransistor ha la base lasciata in circuito aperto; il collettore e l'emettitore sono collegati nel modo classico, con l'emettitore collegato a massa (vedi Fig. 3.9). È consigliabile eseguire alcuni esperimenti variando il valore del resistore, fino ad ottenere un migliore adattamento del circuito. Variate il valore del resistore fino ad ottenere, con il programma presentato nel paragrafo 3.4, adeguate variazioni di valori di ingresso in corrispondenza del campo di sensibilità da voi scelto. Opportune formule di conversione potranno poi calibrare i valori ottenuti.
All'inizio di questo paragrafo abbiamo parlato di risposta spettrale dell'LDR; molti dispositivi optoelettronici si rivelano estremamente sensibili verso l'estremità rossa dello spettro e, in particolare, all'infrarosso. Questo li rende particolarmente idonei per progetti a raggi invisibili co-

me allarmi antifurto o anche impieghi generici a luce bianca. Se il colore della luce è importante (come ad esempio nelle applicazioni fotografiche), il dispositivo in grado di fornire la migliore risposta spettrale è un fotodiodo, il BPW21, che ha una risposta molto simile all'occhio umano. Questo dispositivo può essere usato per misurare intensità luminose (vedi Fig. 3.8); il suo solo svantaggio è il costo, circa il triplo di un ORP12.
È possibile ottenere in maniera più economica un fotodiodo tramite un fototransistor al germanio o al silicio, che in unione ad un resistore da circa 10 kΩ (vedi Fig. 3.8), produrrà variazioni in un campo compreso tra circa 150 (buio) e 20 (luce) all'ingresso dell'ADC. Gran parte dei fototransistor usati con il circuito di base aperto rispondono, similmente all'LDR, all'estremità rossa dello spettro.

3.7 Un sistema di esposizione fotografica

I fotoamatori che usano sviluppare e stampare foto a colori o in bianco e nero potranno utilizzare insieme il misuratore di luce descritto nel paragrafo precedente e un timer in grado di misurare secondi e/o minuti e secondi. È possibile ottenere le letture dal misuratore di luce sullo schermo televisivo, insieme al tempo, nonché far suonare opportuni allarmi dopo tempi richiesti.
Chi desiderasse solamente la visualizzazione del tempo in secondi, potrà usare il seguente programma:

Caricate (LOAD) "espanso" ed eseguite un MERGE con

```
10 POKE 23672,0: POKE 23673,0: POKE 23674,0
20 LET t=INT ((65536*PEEK 23674+256*PEEK 23673+
   PEEK 23672)/50)
30 LET ys=4: LET xs=4: LET cs=8: LET y=100:
   LET d$=STR$t: GOSUB 9390
40 GOTO 20
```

Chi desiderasse tempi espressi in minuti e secondi potrà usare invece il seguente programma:
Caricate "espanso" e fondete con esso il seguente programma:

```
10 LET m=0: LET s=0
15 POKE 23672,0: POKE 23673,0: POKE 23674,0
20 LET t=vedi programma precedente
30 IF s=59 THEN CLS
40 LET m=INT (t/60): LET s=t-(m*60)
```

```
50 LET ys=4: LET xs=4: LET cs=8: LET y=10:
   LET d$=STR$m: GOSUB 9390
60 LET y=100: LET d$=STR$s: GOSUB 9390
70 GOTO 20
```

Desiderando anche allarmi sonori, è possibile aggiungere righe di programma per produrre gli appropriati BEEP. Desiderando, ad esempio, un suono di 90 secondi, potrete inserire nel precedente programma la riga:

```
45 IF m=1 AND s=30 THEN BEEP 1, 10
```

Questa istruzione non farà certo impazzire il timer, che continuerà a seguire il clock del computer. È possibile, se necessario, inserire varie istruzioni di questo tipo.
Chi desiderasse operare a colori potrà adattare la basetta ADC per ottenere sul video, insieme all'indicazione del tempo, la misurazione del contenuto rosso, verde e blu della pellicola. Collegando il circuito misuratore di luce del paragrafo 3.6 alle porte 24, 25 e 26 si potrà sfruttare il seguente programma:

```
   5 LET m=0: LET s=0
  10 POKE 23672,0: POKE 23673,0: POKE 23674,0
  20 LET t=INT ((65536*PEEK 23674+256*PEEK 23
673+PEEK 23672)/50)
  30 IF s=59 THEN  CLS
  40 LET m=INT (t/60): LET s=t-(m*60)
  45 IF s=20 THEN  BEEP 1,10
  50 LET ys=4: LET xs=4: LET cs=8: LET y=10:
LET d$=STR$ m: GO SUB 9390
  60 LET y=100: LET d$=STR$ s: GO SUB 9390
  70 FOR z=1 TO 3
  80 LET ys=2: LET xs=2: LET y=150: LET x=20+
(70*(z-1)): LET d$=STR$ IN (23+z): INK (z+1):
 GO SUB 9400: INK 0
  90 NEXT z
 100 GO TO 20
9390 LET x=(256-xs*cs*LEN d$)/2
9400 LET a=23306: POKE a,x: POKE a+1,y: POKE
a+2,xs: POKE a+3,ys: POKE a+4,cs: LET a=a+4:
FOR i=1 TO LEN d$: POKE a+i,CODE d$(i): NEXT
i: POKE a+i,255: RANDOMIZE USR 32256: RETURN
```

È possibile calibrare opportunamente i valori ottenuti da IN24, IN25 e IN26, per compensare la risposta spettrale del fotorivelatore. In questo caso, il comando STR$ sarà del tipo:

```
STR $ IN (23+z)*w/255
```

È ancora possibile aggiungere allarmi e, in caso di uso del video in una camera oscura fotografica, si provvederà a ridurre opportunamente la luminosità al minimo.
In figura 3.10 possiamo vedere la realizzazione fisica del tipico sistema a 3 colori.

3.8 Una penna ottica

Esistono vari metodi per ottenere sullo schermo disegni e grafici. Si può ad esempio usare direttamente i comandi PRINT AT, PLOT, DRAW,... per tracciare punti sullo schermo, o una penna ottica per disegnarvi direttamente sopra o per segnarvi una posizione per usi futuri. Se, ad esempio, un programma didattico richiedesse allo studente di identificare un oggetto posto sullo schermo, basterebbe piazzare la penna ottica sull'oggetto per rispondere. La penna ottica consiste in pratica nel misuratore di luce descritto nel paragrafo 3.6, incapsulato in un tubo per concentrare la luce in una piccola area di rilevazione, come si vede in figura 3.11.
La penna è collegata all'ingresso 7 della basetta ADC e resa operativa attraverso un opportuno programma di gestione. La procedura di controllo è la seguente:

1. Lo schermo viene reso totalmente nero.
2. La penna viene appoggiata sullo schermo.
3. Viene generata una linea bianca di scansione, che si muove verticalmente lungo lo schermo; quando questa linea incontra la penna, viene rilevata la sua posizione verticale (asse y). Il dato così ottenuto è immagazzinato in memoria.
4. Lo schermo viene reso nuovamente nero.
5. La linea di scansione si sposta ora orizzontalmente sullo schermo, viene rilevata dalla penna, la cui posizione orizzontale (asse x) viene immagazzinata in memoria.
6. Lo schermo ritorna al colore normale, ad esempio bianco, e un opportuno carattere viene visualizzato nella posizione (x, y) occupata prima dalla penna.

Per disegnare sullo schermo, è necessario segnare un insieme di punti:

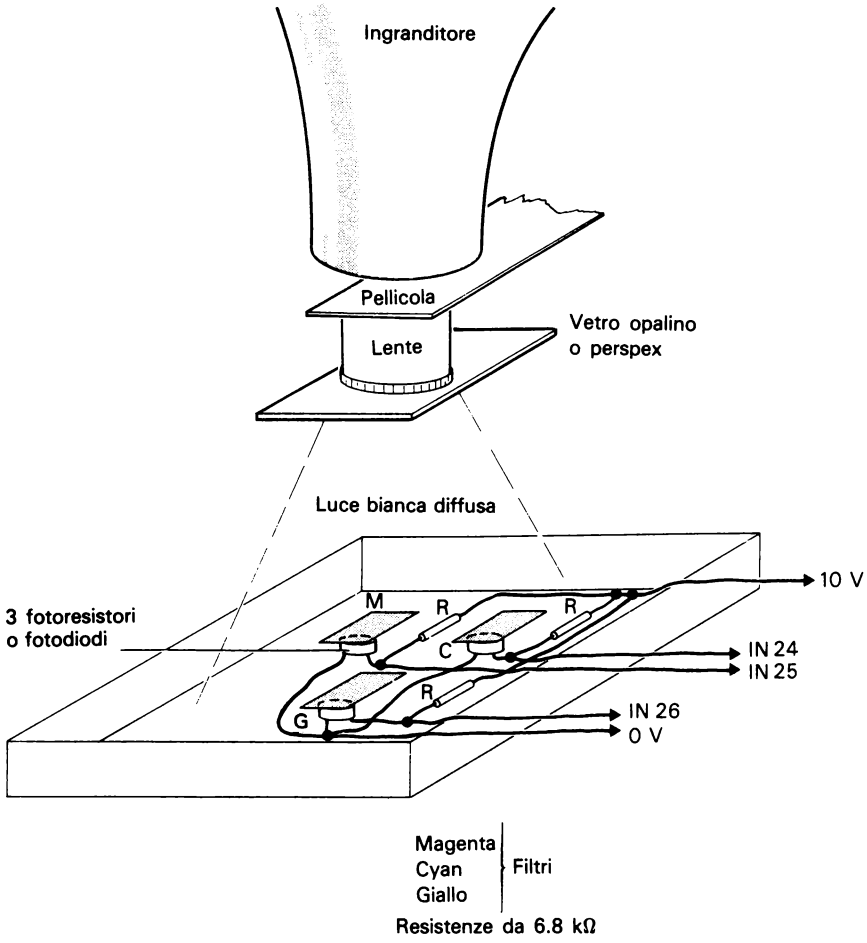


Figura 3.10 Un sistema misuratore di luce per stampa a colori.

una routine in codice-macchina viene usata per salvare il disegno già realizzato in memoria, quindi il processo di scansione rileva il prossimo punto da aggiungere; viene ora visualizzato il disegno originale con l'aggiunta del nuovo punto. La figura 3.12 ci mostra la sequenza di scansione. Formiamo ora due programmi alternativi per realizzare quanto descritto:

1. Un programma BASIC per operare la scansione e gestire la penna.
2. Una routine in codice-macchina.

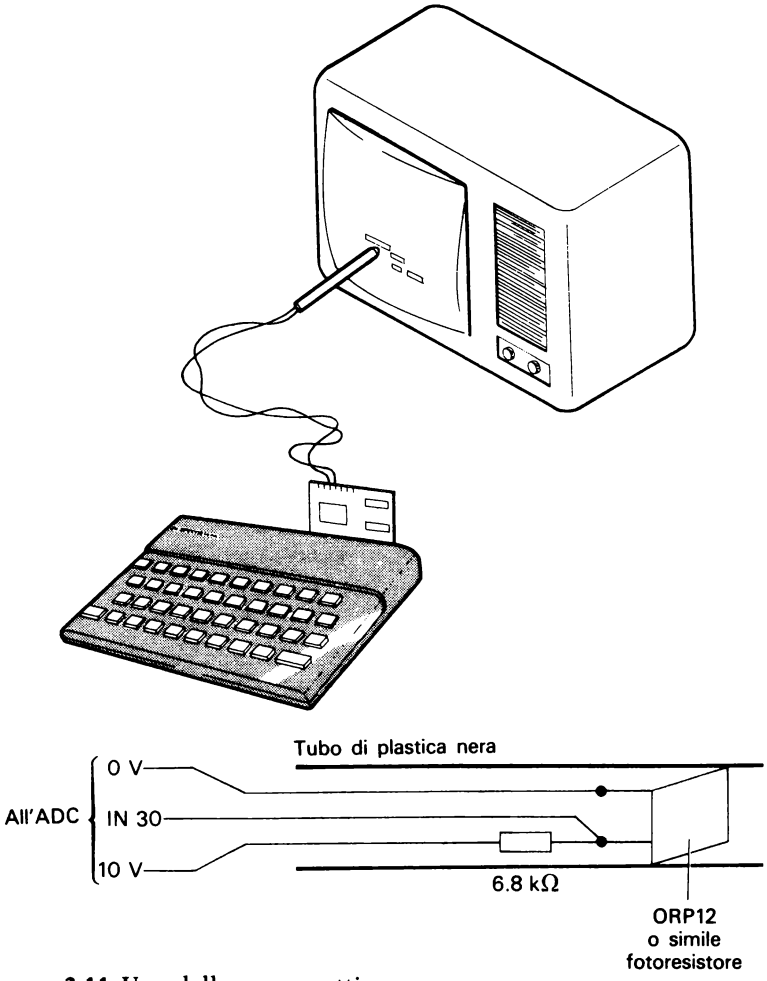


Figura 3.11 Uso della penna ottica.

Il programma BASIC è il seguente:

```
10 CLEAR 44500
15 LET a=0: LET b=0
20 FOR x=0 TO 41
30 READ n
40 POKE (65240+x),n
```

(continua)

```
  50 NEXT x
  60 DATA 17,215,227,33,0,64,6,27,197,6,0,126
,18,19,35,16,249,193,16,244,201
  70 DATA 17,0,64,33,215,227,6,27,197,6,0,126
,18,19,35,16,249,193,16,244,201
  80 CLS : PRINT "Premi qualsiasi tasto": PAU
SE 0
  90 LET l=USR 65240
 100 PAPER 0: INK 7: CLS
 110 FOR x=21 TO 1 STEP -1
 120 PLOT 0,8*x: DRAW 255,0
 130 IF IN 65503<=10 THEN  LET a=x: GO TO 150
 140 NEXT x
 150 CLS
 160 FOR y=1 TO 31
 170 PLOT 8*y,0: DRAW 0,175
 180 IF IN 65503<=10 THEN  LET b=y: GO TO 200
 190 NEXT y
 200 INK 0: PAPER 7: CLS
 210 LET l=USR 65261
 220 PRINT AT (21-a),b;" "
 230 PRINT AT 0,0;"Premi il carattere da dise
gnare"
 240 PAUSE 0: GO TO 90
```

Le linee da 10 a 70 caricano in memoria la routine in codice-macchina necessaria per immagazzinare e restituire il disegno.

La linea 80 costituisce una linea di attesa.
La linea 90 salva in memoria il disegno.
La linea 100 oscura lo schermo.
Le linee da 110 a 140 effettuano la scansione verticale.
Le linee da 160 a 190 effettuano quella orizzontale.
Le linee 130 e 180 controllano, per mezzo dell'indirizzo 30 sull'ADC, la posizione della penna ottica.
La linea 200 riporta al bianco lo schermo.
La linea 210 rimette il disegno sullo schermo.
La linea 220 disegna un □ nel punto di coordinate x, y.
Le linee 230 e 240 richiedono il punto successivo.

Nella figura 3.11 possiamo vedere il circuito e la costruzione della penna.

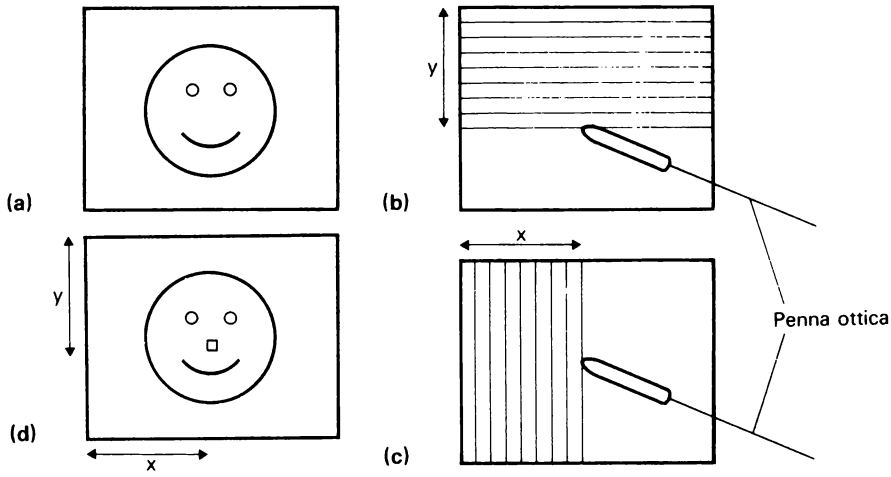


Figura 3.12 Sequenza di scansione per il programma "penna" in codice-macchina. (a) L'immagine sullo schermo viene salvata in memoria. (b) La penna viene posta sullo schermo per essere rilevata dalla scansione verticale. (c) La penna viene piazzata sullo schermo per essere rilevata dalla scansione orizzontale. (d) L'immagine viene posta sullo schermo con l'aggiunta del carattere in posizione (x, y).

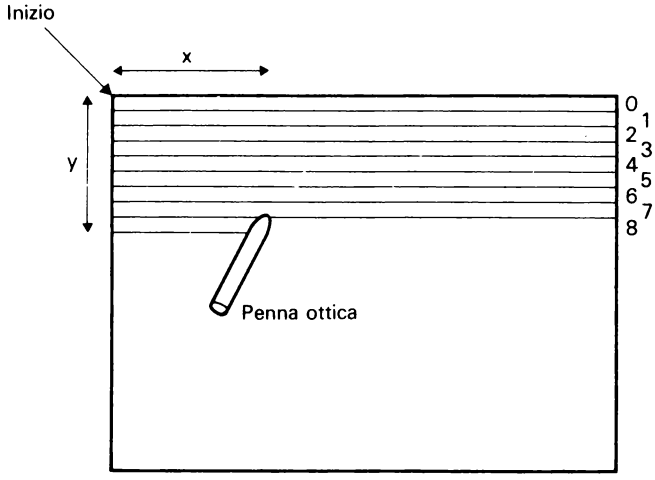


Figura 3.13 Sequenza di scansione per il programma "penna".

I fili dovrebbero essere lunghi circa un metro, il dispositivo fotosensibile dovrebbe essere estremamente piccolo; adatti allo scopo sono, ad esempio, il BPW21 e il BPX65.
Presentiamo ora un programma che, con l'uso del codice-macchina, è più veloce. La procedura è simile a quella seguita dal BASIC, tranne che per il metodo di scansione, mostrato in figura 3.13.

Codifica	*Decimale*	*Risultato*
ld hl,22528	33	(65282)
	0	
	88	
ld c,255	14	
	255	
ld e,22	30	
	22	
ld d,32	22	
	32	
ld(hl),c	113	
in a(30)	219	rivelatore della penna ottica
	30	
sub 40	214	fotorivelatore illuminato?
	40	
jp m, 65311	250	
	31	
	255	
inc(hl)	35	
ld b,255	6	(65300)
	255	
dec b	5	ritardo
jr nz,−3	32	
	253	
dec d	21	

Codifica	*Decimale*	*Risultato*
jr nz−17	32	
	239	
dec e	29	
jr nz−22	32	
	234	
ld c,d	122	(65311)
ld 63744,a	50	
	0	
	249	
ld a,e	123	
ld 63745,a	50	
	1	
	249	
ret	201	(65319)

Le coordinate x e y vengono immagazzinate nelle locazioni di memoria 63744 e 63745 che vengono sfruttate dal programma BASIC per disegnare il □ sullo schermo. È necessario stare attenti a tenere la penna attaccata allo schermo; tenendola infatti distanziata, la penna potrà rilevare un'adiacente linea di scansione, o anche una riflessione della luce ambiente. È pertanto consigliabile operare in ambienti in penombra e ridurre la luminosità del video. È possibile ottenere una scansione completa in pochi millisecondi, ma ciò interferisce con i 50 Hz della frequenza di quadro, producendo quindi solo una parte della scansione. La routine in codice-macchina è rallentata da un ciclo di ritardo caricato nel registro B e, se necessario, il valore 255 può essere variato per aumentare la velocità di esecuzione.
Nel programma BASIC, le linee 60 e 70 sono le routine di salvataggio e restituzione del disegno già usate. La linea 80 è la routine di scansione in codice-macchina appena presentata; il resto si dovrebbe spiegare da sé.
Con poche modifiche, è possibile ottenere la stampa di qualunque carattere dopo ogni scansione. Aggiungete:

```
95 LET a$=" "
```

e sostituite la linea 160 con

160 PRINT AT 0,0; "Premi il carattere che vuoi disegnare":
LET a$=INKEY $:
IF a$=" " THEN GOTO 180: GOTO 100

```
  5 LET p=0: LET z=0
 10 CLEAR 44500
 20 FOR x=0 TO 79
 30 READ n
 40 POKE (65240+x),n
 50 NEXT x
 60 DATA 17,215,227,33,0,64,6,27,197,6,0,126
,18,19,35,16,249,193,16,244,201
 70 DATA 17,0,64,33,215,227,6,27,197,6,0,126
,18,19,35,16,249,193,16,244,201
 80 DATA 33,0,88,14,255,30,22,22,32,113,219,
30,214,40,250,31,255,35,6,255,5,32,253,21,32,
239,29,32,234,122,50,0,249,123,50,1,249,201
 90 CLS : PRINT "Premi qualsiasi tasto": PAU
SE 0
100 LET l=USR 65240
110 PAPER 0: INK 7: CLS
120 LET l=USR 65282
130 INK 0: PAPER 7: CLS
140 LET l=USR 65261
150 PRINT AT 21-(PEEK 63745),31-(PEEK 63744)
;" "
160 PRINT AT 0,0;"Premi il carattere da dise
gnare": PAUSE 0: GO TO 100
```

3.9 Un oscilloscopio digitale a memoria per lo spettro vocale

Questo progetto sfrutta l'ADC per convertire un segnale microfonico in una serie di campioni digitali che vengono immagazzinati in 255 celle di memoria consecutive. Nel capitolo 5 spiegheremo come trasmettere questi dati dal computer al convertitore digitale-analogico per ricostruire il suono originale.

Il programma è il seguente:

```
  10 FOR x=0 TO 28
  20 READ n
  30 POKE (64000+x),n
  40 NEXT x
  50 DATA 17,0,160,245,229,33,255,0,219,30,18
,6,250,5,32,253,19,43,125,183,32,242,124,183,
32,238,225,241,201
  60 PRINT AT 0,0;"parla     ": PAUSE 0
  65 PRINT AT 0,7;">"
  70 LET l=USR 64000
  80 INK 0: PAPER 7: BORDER 7: CLS
  90 FOR x=0 TO 255
 100 PLOT x,0: DRAW 0,(PEEK (40960+x))
 110 NEXT x
 120 PRINT AT 0,0;"Premi un tasto per continu
are": PAUSE 0
 130 CLS : GO TO 60
```

Il programma in codice-macchina della linea 50 è il seguente:

Codifica	*Decimale*	*Risultato*
ld de,40960	17	
	0	indirizzo iniziale del buffer
	160	
push af	245	
push hl	229	
ld hl,	33	
255	255	numero di campionamenti
	0	
in a,(30)	219	
	30	acquisisce il segnale microfonico
ld (de),a	18	lo salva in memoria

Codifica	*Decimale*	*Risultato*
	6	aggiusta il tempo fra i campionamenti
	250	
	5	
	32	
	253	
inc de	19	campionamento successivo
dec hl	43	diminuisce HL e termina il ciclo quando HL=0
ld a,l	125	
or a	183	
jr nz,−9	32	
	242	
ld a,h	124	
or a	183	
jr nz,−13	32	
	238	
pop hl	225	
pop af	241	
ret	201	

Questo programma visualizza una traccia oscilloscopica a barre verticali sullo schermo. All'apparizione della parola "parla", premete un tasto e generate un suono nel microfono per circa un secondo. Vedrete immediatamente visualizzato sullo schermo lo spettro del segnale sonoro; la locazione 40960 e le 255 successive conserveranno i campioni di suono per ogni possibile uso futuro. Ovviamente il prossimo campionamento cancellerà i dati precedenti e li rimpiazzerà con quelli appena acquisiti. Desiderando ottenere una visualizzazione per punti, sostituite la linea 100 con:

```
100 PLOT x, PEEK (40960+x)
```

Servirà un amplificatore audio per portare il piccolo segnale di uscita dal microfono al livello (di alcuni volt) necessario per pilotare l'ingresso dell'ADC. Se avete costruito la basetta DAC, potrete sfruttare l'amplificatore incluso in essa; comunque, sarà sufficiente qualunque amplificatore audio con alta impedenza di uscita.

3.10 Un termometro

Nella figura 3.14a possiamo vedere il circuito di un termometro estremamente semplice, che sfrutta il termistore RS151-257, la cui curva di taratura (nel campo da −80°C a 150°C) è presentata in figura 3.14b. Il segnale di ingresso alla basetta ADC sarà variabile da un valore estremamente prossimo a 0 (quando, a bassissime temperature, il termistore avrà una resistenza di vari MΩ), a circa 10 V alla temperatura di 150°C.

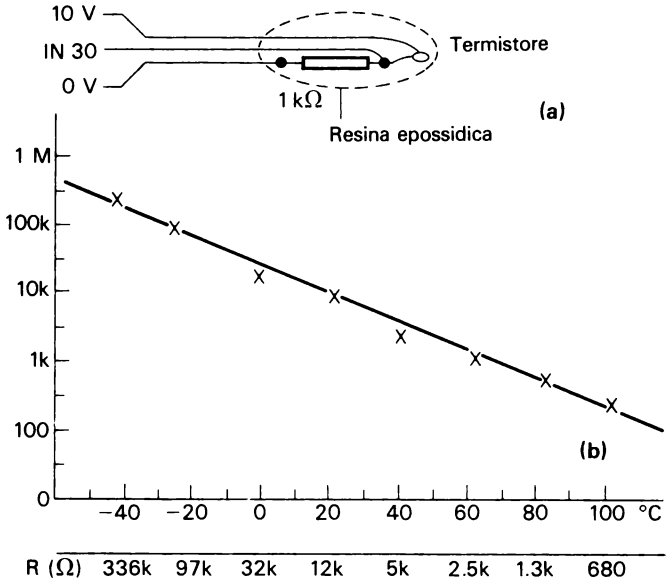


Figura 3.14 (a) Un circuito termometrico. (b) La curva di taratura.

Il termistore è un componente molto delicato, e andrà incapsulato, insieme ad una resistenza da 1 kΩ, in una "camicia" di resina epossidica (Araldite) come si vede in figura 3.15. Questo vi permetterà di immergere il sensore in liquidi e gas per ottenere misurazioni di temperatura.
Il programma BASIC viene presentato sotto. È possibile ottenere allarmi per temperature eccessivamente alte o basse, inserendo un'istruzione come:

```
92 IF t<10 OR t>30 THEN BEEP 1, 10
```

Caricate il programma "espanso" e fatelo girare, poi caricate quello riportato qui a fianco ("temp") ed eseguite RUN.

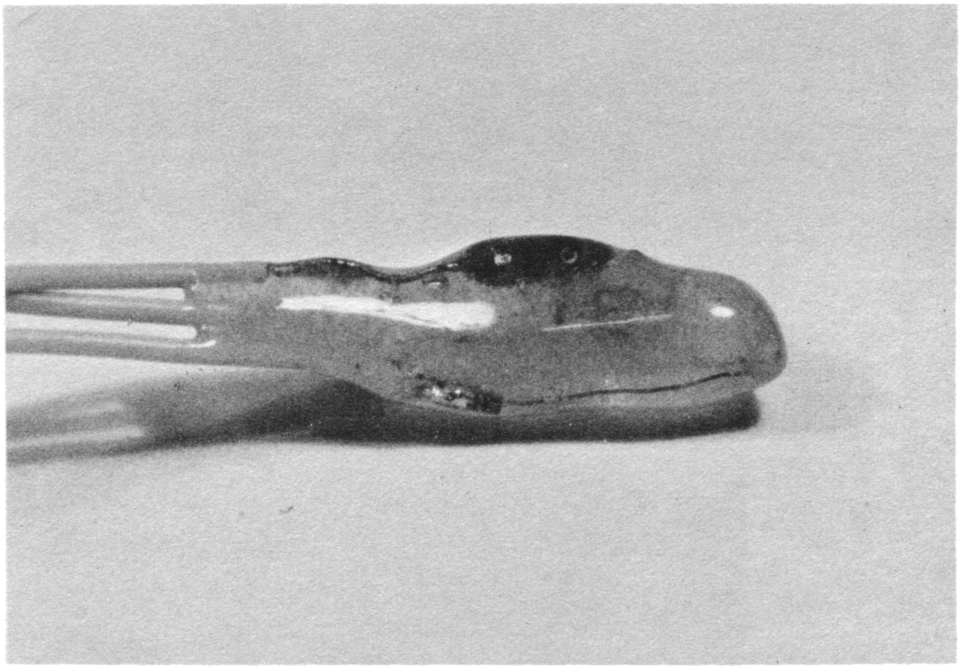


Figura 3.15 Termistore incapsulato in resina epossidica. Il termistore è visibile nell'angolo inferiore destro.

```
   5 LET oldt=0
  80 PRINT AT 0,0;"temp C"
  90 LET t=IN 30-15
  91 IF oldt<>t THEN  CLS
  95 LET oldt=t
 100 LET ys=4: LET xs=4: LET y=100: LET d$=ST
R$ t: GO SUB 9390: GO TO 80
9390 LET x=50: LET cs=8
9400 LET a=23306: POKE a,x: POKE a+1,y: POKE
a+2,xs: POKE a+3,ys: POKE a+4,cs: LET a=a+4:
FOR i=1 TO LEN d$: POKE a+i,CODE d$(i): NEXT
i: POKE a+i,255: RANDOMIZE USR 32256: RETURN
```

Capitolo

Conversione digitale-analogica

4

Nel mondo esterno al computer i segnali sono analogici, all'interno del computer i segnali sono digitali. Nel capitolo 3 abbiamo presentato la prima interfaccia tra il mondo esterno e il computer stesso. Questo capitolo farà l'opposto, convertendo in impulsi o segnali direttamente utilizzabili, i molti segnali digitali che possono essere generati dal computer o presenti nella sua memoria.
Vengono descritte due schede principali:

1. La scheda latch che tratta le parole da 8 bit del computer e le trasmette in una serie di 8 linee parallele i cui valori possono essere 0 V o 5 V.
2. Il convertitore digitale-analogico, la cui basetta contiene, oltre alla circuiteria latch anche un chip di conversione per generare i 256 numeri, da 0 a 255 corrispondenti a ciascuna parola da 8 bit.

4.1 La scheda latch

Il chip di latch è il 74LS374 che è in grado di accettare i segnali del bus dati dello Spectrum quando e solo quando esso è abilitato dall'ingresso CLOCK. Questo chip conserva lo stato fino all'arrivo del successivo impulso di CLOCK. Questo tipo di funzionamento viene detto latch (serratura a scatto). Se non si usasse questo chip l'uscita sarebbe una serie senza senso di impulsi ad alta frequenza su ognuna delle linee dati dello Spectrum. La scheda latch è mostrata in figura 4.1. L'indirizzamento è generato da due porte del 74LS27, in modo simile a come già fatto nel sistema

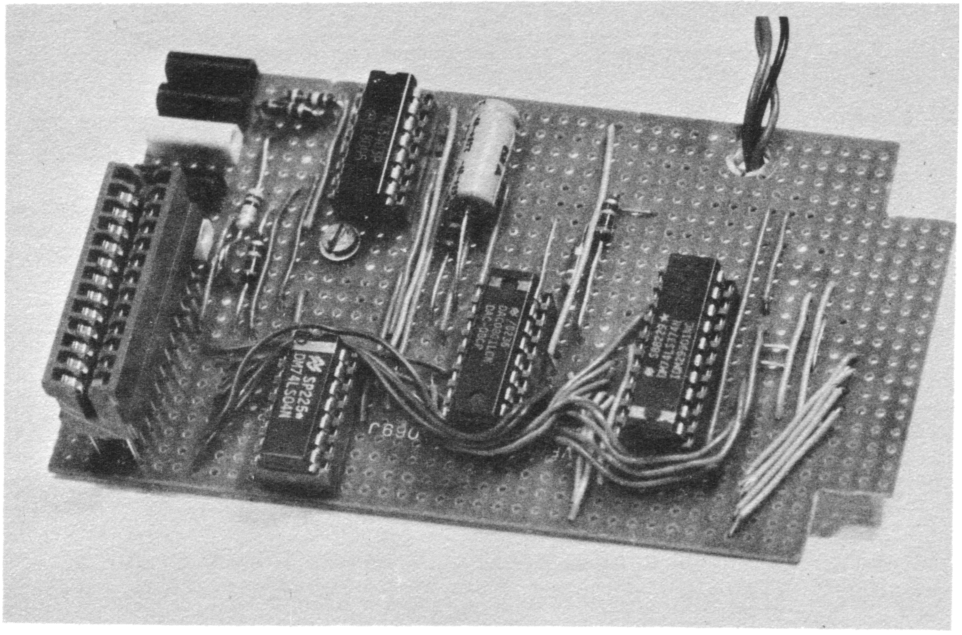


Figura 4.1 Basetta del convertitore digitale-analogico. Da notare i collegamenti per l'alimentazione, i connettori da 2 mm e il connettore a pettine per la scheda latch relè.

ADC presentato nel capitolo 3. Abbiamo scelto ancora l'indirizzo 31 poiché per questo basta decodificare un singolo segnale sulla linea d'indirizzo A5 insieme agli stati delle linee $\overline{WR}$ (la linea di scrittura) e $\overline{IORQ}$ (la linea che segnala un'operazione di ingresso-uscita). Un gate controlla la situazione di dette linee e l'altro inverte l'uscita del precedente per abilitare il chip di latch. Se, per esempio, la locazione di memoria 64000 contenesse il numero 255 e noi volessimo comandare da questa locazione dei circuiti connessi allo Spectrum, dovremmo agire secondo la seguente procedura:

1. Il programma BASIC deve provvedere a generare in uscita il numero 255:
   10 OUT 31, PEEK 64000
2. L'equivalente binario di 255 (cioè 11111111) apparirà ora sul bus dati e sui piedini corrispondenti dell'integrato 74LS374 (vedi Fig. 4.2).
3. La combinazione dei segnali A5, RD, e IORQ abilita il 74LS27 e fa sì che esso blocchi a 1 le uscite dei suoi otto flip-flop per trasmettere otto 1 ai piedini di uscita.

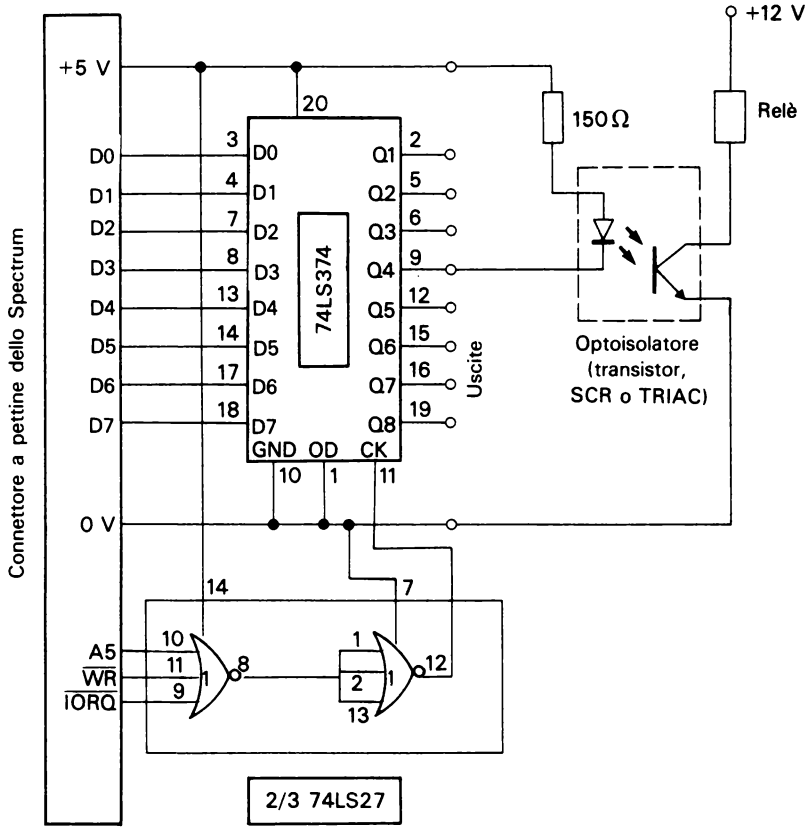


Figura 4.2 Il circuito elettrico del latch.

4. Il programma continuerà con l'esecuzione della successiva linea BASIC che rimuoverà i segnali di indirizzamento.
5. Gli otto 1 emessi resteranno comunque presenti sulle uscite del latch fino all'arrivo del prossimo comando OUT 31.

Lo schema elettrico del circuito presentato in figura 4.2 è molto semplice e dovrebbe spiegarsi da sé. Si possono sfruttare le uscite TTL per pilotare direttamente dei LED, ma è preferibile usare degli optoisolatori che separino la delicata circuiteria della scheda latch e lo Spectrum dal mondo esterno. In figura 4.3 possiamo vedere cinque applicazioni di optoisolatori che usano i tre seguenti dispositivi fondamentali:

1. Un optotransistor con coppia Darlington e LED. Questo tipo di componente può essere usato per pilotare direttamente un relè, una piccola

lampadina oppure un circuito ad alto voltaggio attraverso un triac o un tiristore.
2. Un optotiristore per semplici circuiti di alimentazione a semi-onda come comandi idraulici di valvole, pompe, luci e motori che richiedano semplicemente comandi di accensione e spegnimento.
3. Un circuito optodiac/optotriac che può pilotare semplici circuiti di alimentazione direttamente, come nel secondo dispositivo, o attraverso un triac per applicazioni che richiedono una maggiore potenza.

La scheda latch può pertanto essere connessa ad otto dispositivi optoelettronici come quelli appena presentati, purché la tensione di alimentazione 5 V sia in grado di erogare adeguata potenza. I circuiti esterni dovrebbero disporre di propri alimentatori, così da non sovraccaricare l'alimentatore dello Spectrum.
Un altro dispositivo elettronico che può essere con ampia sicurezza usato come interfaccia tra Spectrum, latch o DAC e i circuiti esterni in continua a bassa tensione è il transistor V-FET o VMOS. Si tratta di un transistor ad effetto di campo ad alta corrente in grado di funzionare molto bene come interruttore di potenza per pilotare motori in continua, (come nei progetti di controllo di trenini che incontreremo nel paragrafo 5.6) e servomeccanismi. I tiristori e i triac sono dei dispositivi in grado di lavorare in corrente alternata ma non di aprire il circuito se alimentati in continua, mentre in tali condizioni, il V-FET, col carico connesso in serie al circuito di drain o di source, aprirà e chiuderà il circuito esattamente come un relè. Un V-FET tipico è il modello VN46AF in grado di sopportare una corrente fino a 2 A e una tensione di alimentazione massima di 40 V; si tratta di un transistor da 15 W.
Nella figura 4.3 possiamo vedere questo dispositivo e un suo tipico circuito di applicazione.

4.2 Costruzione della scheda latch

È disponibile una basetta a circuito stampato già pronta. In alternativa è possibile costruire il circuito sulla basetta millefori SRBP (vedi App. H). In figura 4.4 possiamo vedere la realizzazione della scheda latch. Si tratta di un lavoro molto semplice, che richiede, a differenza dell'ADC o del DAC, un solo filo isolato in PVC, e vari fili di rame nudo.
Il connettore a pettine viene cablato come si vede nelle figure e i suoi piedini vengono tagliati in modo analogo a quanto già fatto per la scheda ADC e montato sulla basetta in modo che sia possibile la sua massima rigidità. All'altro estremo della basetta, dal lato componenti, troveremo l'uscita del latch, che è progettata in modo tale da accettare — attraverso

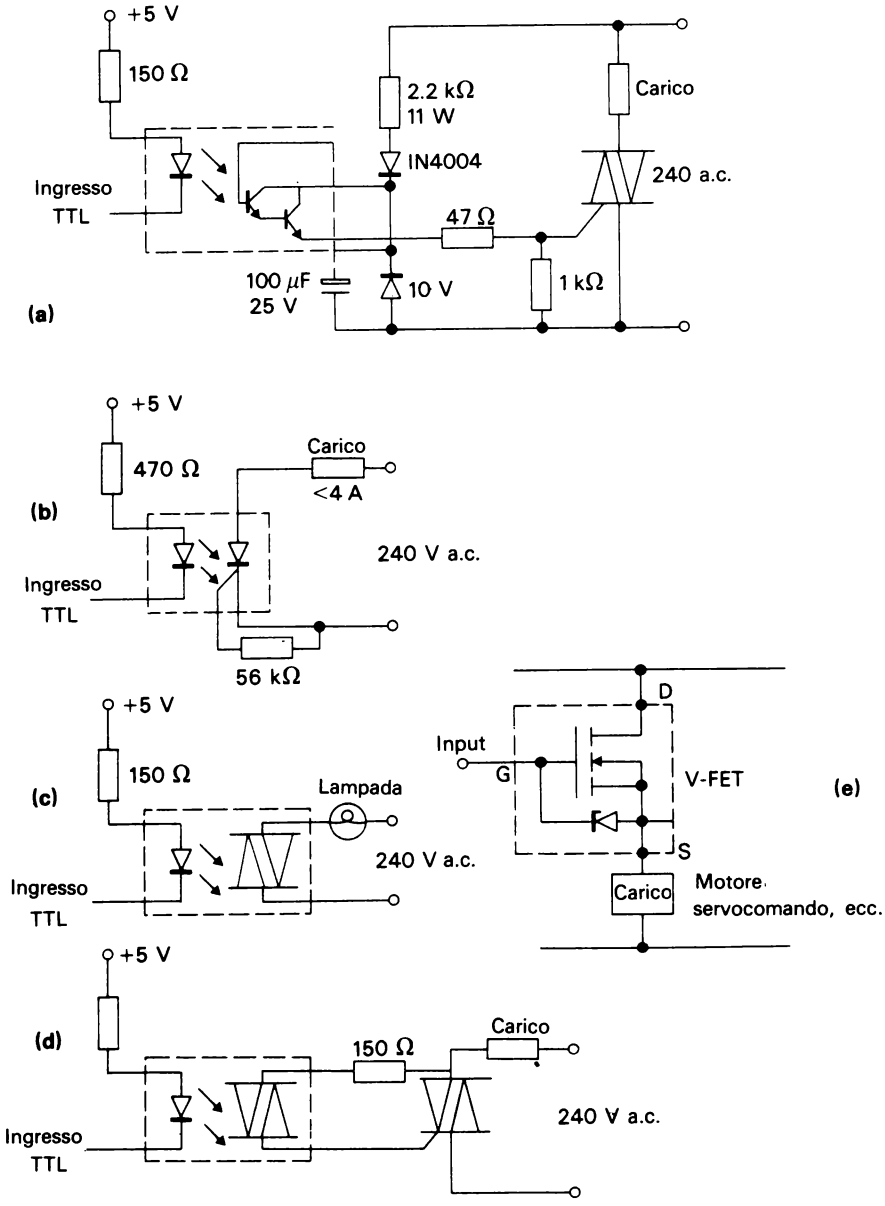


Figura 4.3 Cinque circuiti di isolamento per interfacciare il 74LS374 col mondo esterno. (a) Coppia optodarlington. (b) Optotiristore. (c) Optotriac. (d) Optotriac che pilota un triac di maggiore potenza. (e) Un V-FET.

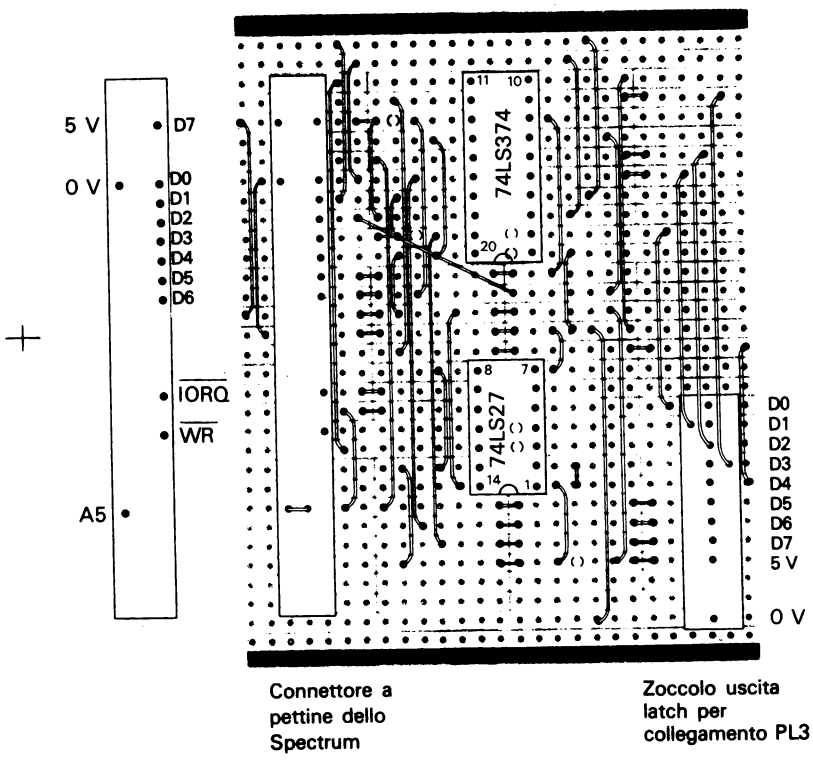


Figura 4.4 Realizzazione della basetta latch.

il connettore PL3 — una basetta di optoisolatori. È possibile accoppiare dispositivi alla scheda latch in altri modi; il montaggio della scheda relè è quindi opzionale. È possibile anche montare direttamente sulla scheda latch una serie di optoisolatori.

In figura 4.5 possiamo vedere una scheda che usa gli optotransistor mostrati in figura 4.3a per controllare dei piccoli relè la cui eccitazione sia ad una tensione compresa tra 5 V e 8 V. Questi relè a loro volta potranno controllare le luci per alberi di Natale, luci per discoteca, pompe ed altri dispositivi (vedi i paragrafi 4.7 e 4.10). La scheda, se usata con optotiristori o optodiac, può essere tagliata lungo la linea x-x e le connessioni di rete possono essere saldate direttamente alla basetta. PRESTATE LA MASSIMA ATTENZIONE ALLA TENSIONE DI RETE. Essa costituisce infatti un pericolo non solo per l'utente ma anche per tutta la parte elettronica. Le connessioni dei relè mostrati in figura 4.5 non comprendono il collegamento dei contatti poiché questo sarà progettato in funzione delle

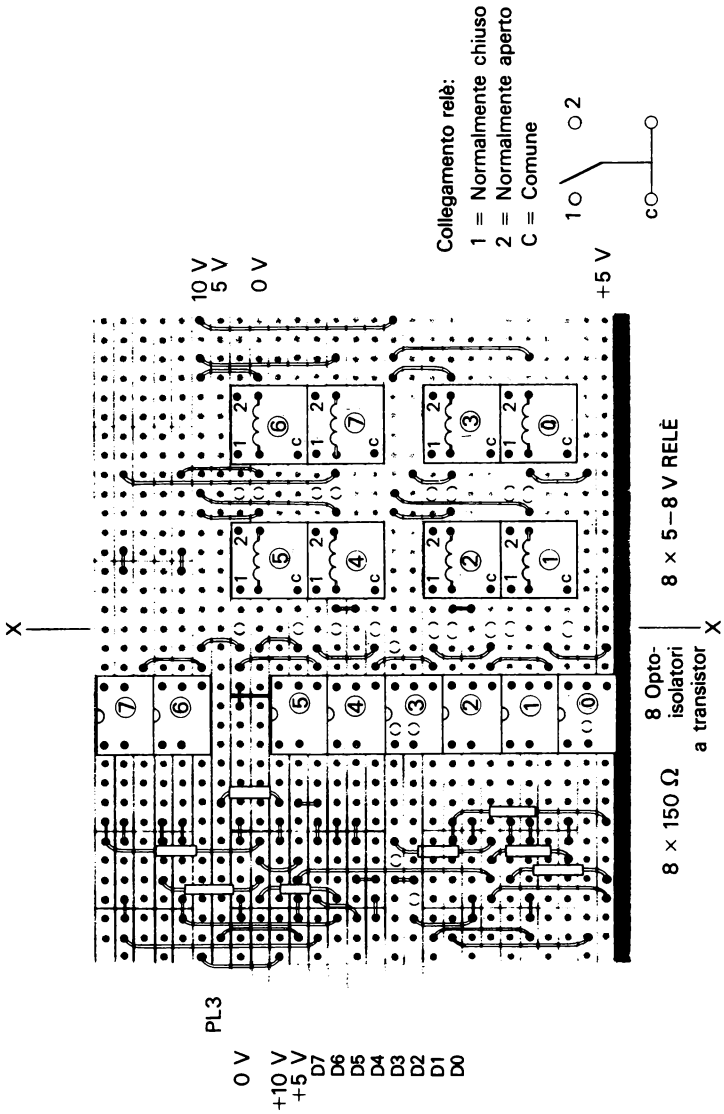


Figura 4.5 La scheda latch / optoisolatori / relè.

applicazioni che potrebbero anche rendere necessarie interconnessioni fra i vari relè.
Tanto la scheda latch (Fig. 4.4) che la scheda relè (Fig. 4.5) usano la basetta tipo RS tagliata a misura e preparata come si vede nelle figure: come già nel capitolo precedente, le tracce tagliate sono marcate ( ).

4.3 Collaudo della scheda latch

Ogni connessione di schede allo Spectrum dovrà ovviamente avvenire con le alimentazioni spente. Non rispettando questa semplice regola i transistor ed eventuali difetti possono generare danni permanenti tanto al vostro computer che alle schede.
Controllate attentamente la scheda, servendovi prima della vista e poi di un ohmmetro per ricercare cortocircuiti ed errori di cablaggio. Quando ritenete che la scheda sia a posto, inseritela nello Spectrum e fornite alimentazione. Lo Spectrum dovrebbe agire nel modo ormai usuale, ma, se così non fosse, spegnete l'alimentazione e ricontrollate la scheda più attentamente.
Le tensioni di uscita sui terminali D0-D7 del connettore SK3 dovrebbero essere tutte a 0 logico (circa 0 V). Digitate:

OUT 31,255 (seguito da ENTER)

e tutte le uscite assumeranno il valore logico 1 (circa 5 V). Spegnete ora lo Spectrum e inserite nel connettore SK3 (attenzione: questo connettore non è dotato di chiave di polarizzazione) la scheda relè o la scheda optoisolatori e riaccendete il computer. Gli optoisolatori dovrebbero funzionare contemporaneamente alle uscite del latch. I relè e gli altri dispositivi dovranno disporre di una tensione di alimentazione separata (10 V come in figura 4.5) e l'intero sistema dovrebbe essere ora collaudato usando le ormai usuali tecniche.

4.4 Convertitore digitale-analogico

La seconda scheda fondamentale che presentiamo in questo libro contiene un completo convertitore digitale-analogico ad otto bit dotato di un latch integrato (simile a quello appena descritto) e di amplificatori di ingresso e di uscita. L'amplificatore di ingresso usato in questa scheda è uno dei quattro amplificatori operazionali contenuti nel 3403, che può venire sfruttato per pilotare qualunque ingresso dell'ADC descritto nel ter-

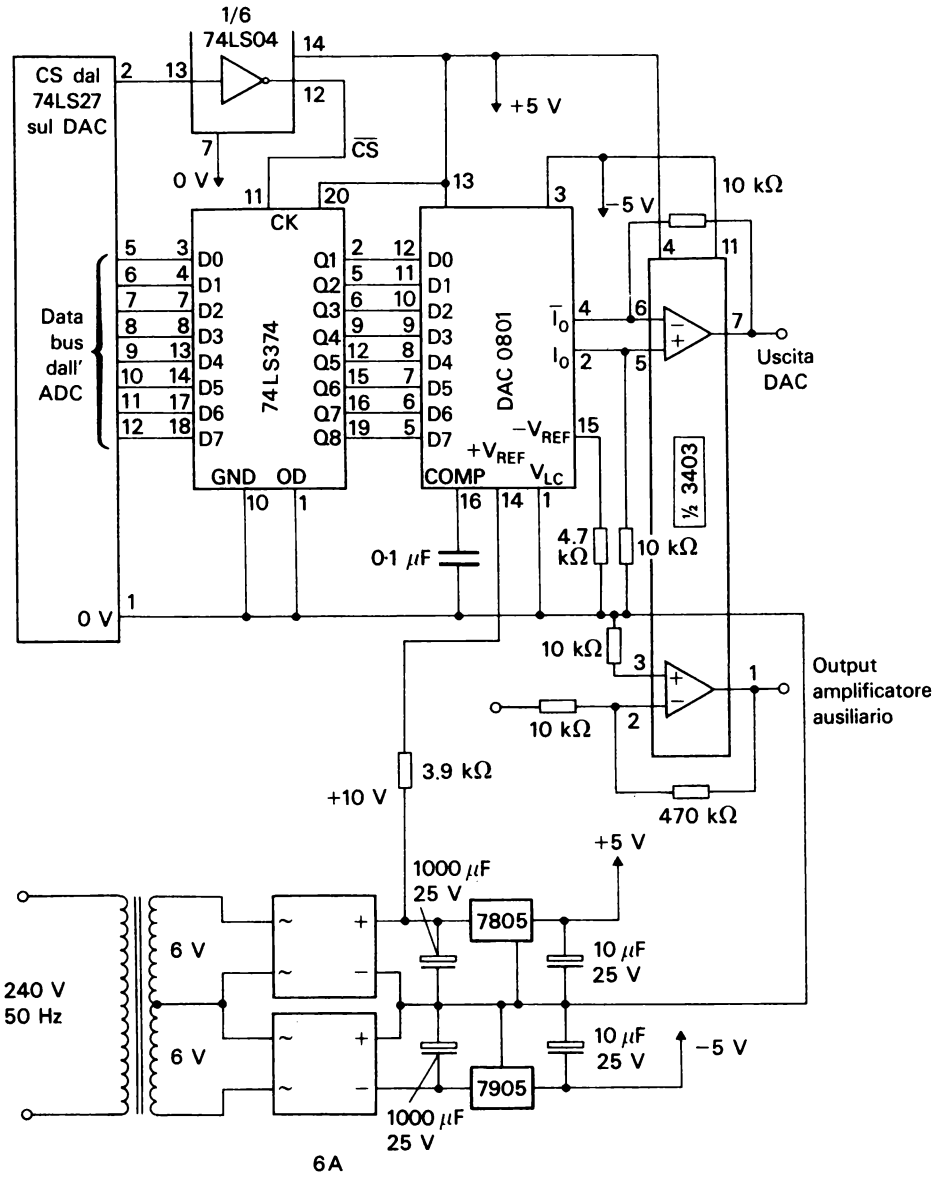


Figura 4.6 Schema elettrico del convertitore digitale-analogico e relativo alimentatore.

zo capitolo. I due amplificatori rimanenti in questo integrato possono, se necessario, essere connessi lungo le linee dei piedini 1, 2 e 3. La piedinatura completa del 3403 è presentata nell'Appendice E.
Lo schema elettrico del DAC è mostrato in figura 4.6. Questa scheda è stata progettata in modo tale da essere direttamente inserita nel connettore posto sul retro della basetta ADC, da cui saranno prelevati gli opportuni segnali. Si è fatta questa scelta poiché molte applicazioni usano tanto l'ADC che il DAC e questo tipo di connessione è sembrato il più conveniente. Richiedendo ben tre tensioni di alimentazione (+5 V, −5 V e +10 V), il DAC non sfrutta l'alimentatore dello Spectrum: ho tentato, senza successo, di prelevare queste tensioni di alimentazione dallo Spectrum. Sarà pertanto necessario usare per questa basetta un semplicissimo alimentatore (vedi Fig. 4.7).
Come convertitore digitale-analogico si è scelto il modello DAC0801. Il circuito di uscita è stato progettato in modo tale da poter pilotare direttamente gli ingressi invertente e non invertente di un amplificatore operazionale, generando pertanto un segnale di uscita centrato intorno a 0 V. Il latch è simile a quello già presentato. Il circuito di decodifica dell'indirizzo è quello contenuto sulla basetta ADC, salvo l'inversione del segnale CS operata da uno dei sei inverter contenuti nel 74LS04 sulla basetta DAC.

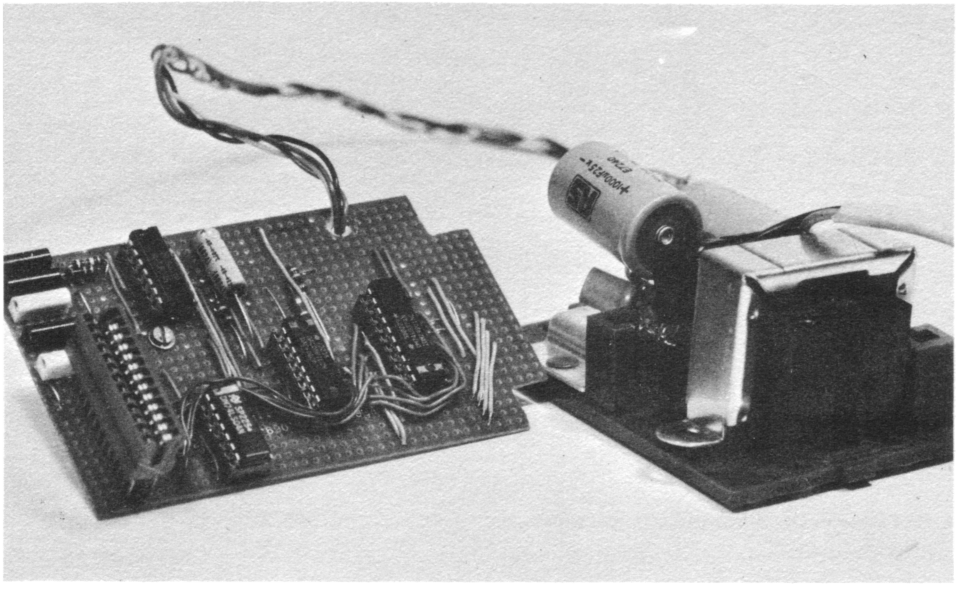


Figura 4.7 Basetta DAC con il suo alimentatore.

Questo circuito converte qualunque parola binaria a 8 bit direttamente in un numero compreso tra 0 e 255 usando una tecnica (a scala di resistori) comune nei convertitori digitali-analogici. Essendo la conversione istantanea, non è necessaria alcuna temporizzazione e la circuiteria è pertanto abbastanza semplice.

4.5 Costruzione del DAC

Anche per questo è disponibile una basetta a circuito stampato; in alternativa è possibile usare una basetta millefori tipo SRBP (vedi App. H). A prima vista, lo schema di cablaggio presentato in figura 4.8 sembra un po' complesso. Il cablaggio, tuttavia, è relativamente semplice ed è stato rappresentato in quel modo esclusivamente per motivi di chiarezza. In effetti, i fili isolati in PVC seguiranno il percorso più breve fra i punti che devono connettere e sarà possibile costruire il tutto in maniera abbastanza compatta e facile.
Tanti ponticelli con filo isolato sono resi necessari dal "disordine" nella piedinatura dei circuiti integrati. Lo Spectrum è dotato di otto linee dati, che terminano su vari piedini e non sono in ordine. Le linee D0-D7 del 74LS374 e del DAC 0801 sono disposte in modi completamente differenti ed è pertanto necessario ricorrere ad un cablaggio da piedino a piedino. Sarebbe possibile ridurre questa complessità di cablaggio montando il tutto su una basetta più grande, che dovrebbe però essere di dimensioni quadruple. Ad ogni modo, questa scheda funziona. Per costruire la basetta usate la procedura ormai classica:

1. Tagliate la basetta a misura e interrompete le tracce di rame nei punti segnati con ( ).
2. Inserite gli zoccoli per i circuiti integrati.
3. Inserite i ponticelli di filo, gran parte dei quali in filo isolato in PVC da 0.6 mm.
4. Inserite gli altri componenti e i connettori.
5. Costruite l'alimentatore e attaccate direttamente a questa basetta i fili di connessione lunghi circa 20 cm.
6. Inserite i circuiti integrati.

4.6 Collaudo del DAC

Eseguite i consueti controlli. Con la basetta ADC completamente provata e funzionante, spegnete lo Spectrum, inserite la basetta DAC e riaccende-

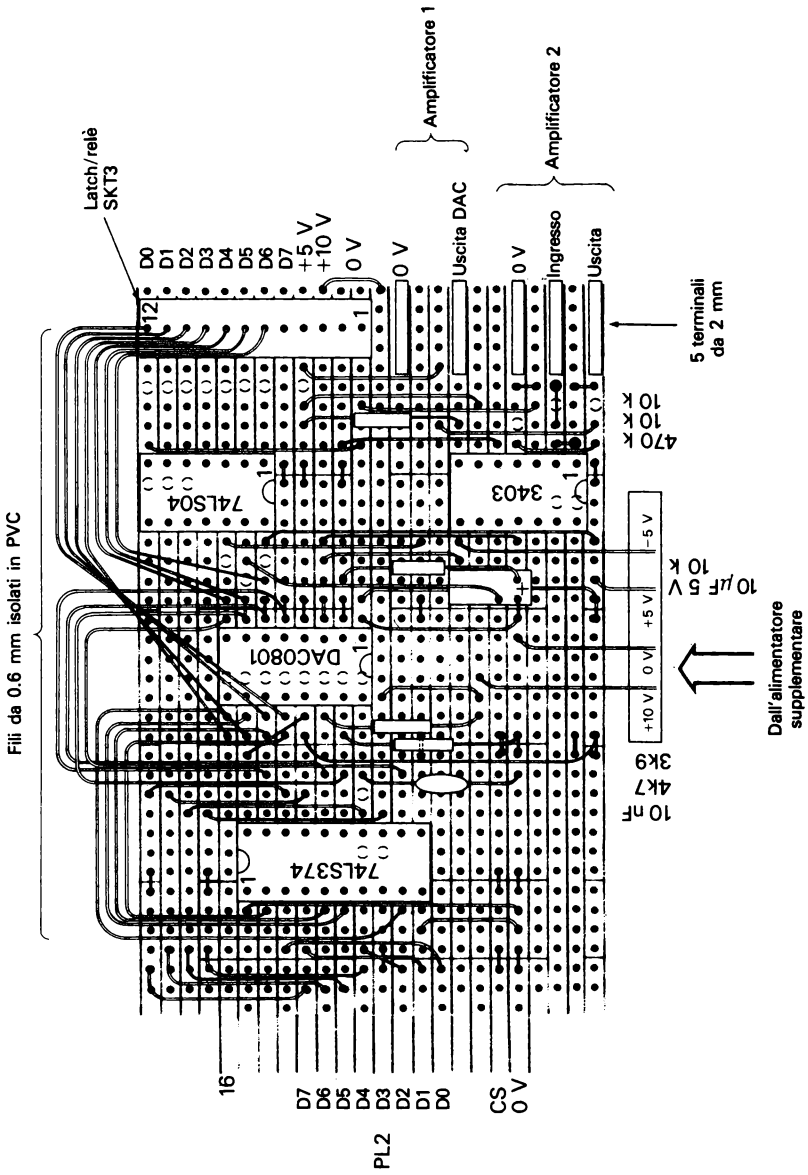


Figura 4.8 Realizzazione della basetta DAC.

te il computer. Per ora, non alimentate la basetta DAC. Se il computer funziona normalmente, dovrebbe essere tutto a posto.
A parte provvedete a provare l'alimentatore che avete appena costruito e controllate le tensioni di +5 V, −5 V e +10 V.
Spegnete ora lo Spectrum, connettete anche questo nuovo alimentatore e fornite alimentazione al tutto. Ancora una volta lo Spectrum dovrebbe funzionare normalmente. In figura 4.9 possiamo vedere le basette ADC e DAC collegate allo Spectrum. Digitate:

```
10 FOR x=1 TO 255
20 OUT 31, x
30 NEXT x
40 GOTO 10
```

Questo genererà una "rampa" di uscita agevolmente riscontrabile collegando un voltmetro fra l'uscita DAC e 0 V. La rampa durerà circa un secondo alla volta. Attraverso un oscilloscopio potremo agevolmente vedere i 255 gradini che compongono questa forma d'onda. Infine, per provare l'amplificatore operazionale dell'ADC usate un microfono e il progetto descritto nel paragrafo 3.9.

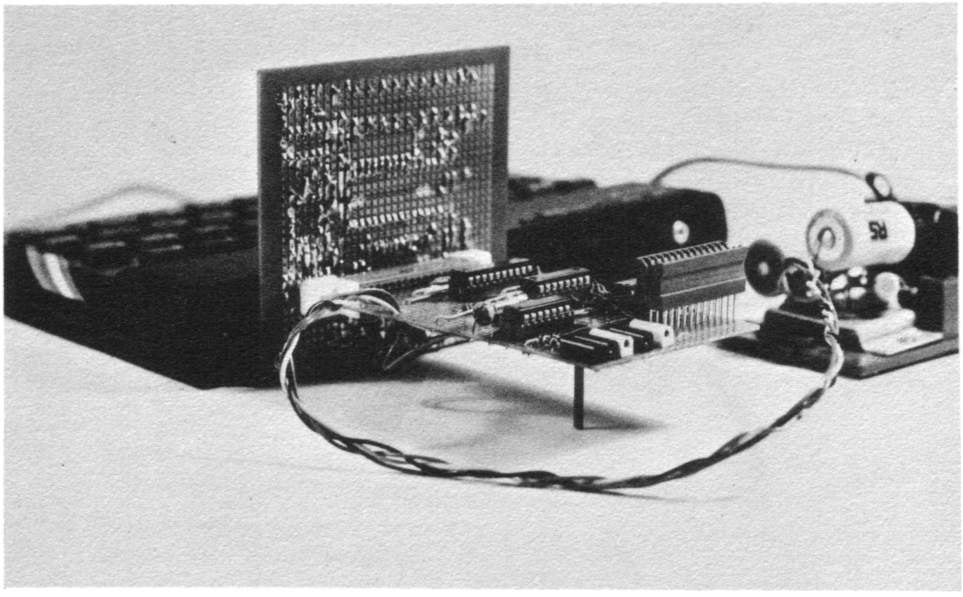


Figura 4.9 Basetta DAC inserita nella scheda ADC connessa allo Spectrum. Notare il distanziatore tipo 6BA necessario a reggere il peso della basetta DAC e della scheda latch.

4.7 Luci per l'albero di Natale o per la discoteca

Questo progetto si basa su un timer discreto (che determina cioé, l'intervallo tra due eventi) che trova svariate applicazioni. Il listato del programma è il seguente:

```
  10 DIM a(8): DIM b(100): DIM p(100): DIM q(
10)
  20 LET z=0: LET y=0: LET g=0
 100 INPUT "numero degli eventi  ";a: LET g=a
 105 PRINT g
 110 PRINT "evento          periodo"
 120 FOR n=1 TO a
 130 INPUT "batti un numero ad 8 bit      ";b
(n)
 140 PRINT n
 150 INPUT "batti l'intervallo (max 255)  ";p
(n)
 160 PRINT TAB 6;b(n);TAB 20;p(n)
 170 LET z=b(n)
 180 GO SUB 230
 190 POKE (40000+2*n),y
 200 POKE (40001+2*n),p(n)
 210 NEXT n
 220 PRINT "premi un tasto per iniziare": PAU
SE 0: GO TO 300
 230 LET y=0
 235 FOR v=8 TO 1 STEP -1
 240 IF INT ((10^(v-1))-z)<=0 THEN  LET q(v)=
2^(v-1): GO TO 260
 250 IF INT ((10^(v-1))-z)>0 THEN  LET q(v)=0
: GO TO 270
 260 LET z=(z-(10^(v-1)))
 270 LET y=y+q(v)
 280 NEXT v
 290 RETURN
 300 FOR b=1 TO g
 310 LET y=PEEK (40000+2*b)
 320 OUT 31,y
 330 GO SUB 500
 340 PRINT "evento ";b;"  ";z
```

(continua)

```
 350 PRINT TAB 17;"periodo ";b;"   ";PEEK (40
001+2*b)
 360 PAUSE PEEK (40001+2*b)
 370 NEXT b
 380 STOP
 500 LET z=0
 505 FOR w=8 TO 1 STEP -1
 510 IF INT (y-(2^(w-1)))<0 THEN  LET a(w)=0:
 GO TO 540
 520 IF INT (y-(2^(w-1)))>=0 THEN  LET a(w)=1
0^(w-1)
 530 LET y=(y-(2^(w-1)))
 540 LET z=z+a(w)
 550 NEXT w
 560 RETURN
1000 FOR x=0 TO 15
1010 PRINT PEEK (40000+x)
1020 NEXT x
```

```
evento          periodo
1
      11100011      50
2
      10000100      30
3
      10            150
premi un tasto per iniziare
evento 1  11100011
                periodo 1   50
evento 2  10000101
                periodo 2   30
evento 3  10
                periodo 3   150
```

Il programma si compone di varie parti:

1. Le linee 10-170 permettono all'utente di stabilire gli intervalli e di dettagliare gli eventi.
2. Le linee 190 e 200 inseriscono (POKE) questi dati in locazioni di memoria consecutive dove resteranno per usi successivi.
3. Le linee 230-290 sono una subroutine di conversione da binario a decimale che permette all'utente di programmare gli eventi direttamente sotto forma binaria a 8 bit. Purtroppo non è possibile usare in questo contesto il comando BIN: è possibile solo incorporarlo in un programma.
4. Le linee 300-380 prelevano ordinatamente (PEEK) i dati dalle locazioni di memoria, evento per evento, e li trasmettono alla scheda latch.
5. Le linee 500-560 costituiscono una subroutine di conversione da decimale a binario in modo da visualizzare sullo schermo gli 0 e gli 1. Facciamo ora un esempio di funzionamento di un semaforo:

Evento	*Intervallo*	R	G	V	R	G	V	*(riserva)*	
1	200	1	0	0	0	0	1	0	0
2	200	1	0	0	0	1	0	0	0
3	200	1	1	0	1	0	0	0	0
4	200	0	0	1	1	0	0	0	0

Il blocco di semafori dotati di luci rosse (R), gialle (G) e verdi (V) commuterà le luci con uguali intervalli (200) di tempo. I dati del programma sopra esposto saranno introdotti come segue:

Numero degli eventi	4
Intervalli/numero ad 8 bit	10000100
	200
	10001000
	200
	11010000
	200
	00110000
	200

Il programma di comando inizia alla linea 300 e, via via che accadono i vari eventi, il numero binario viene visualizzato sullo schermo contempo-

raneamente alla sua trasmissione al circuito di latch. I relè, le luci o gli altri dispositivi si accenderanno e spegneranno insieme ai dati. È possibile semplificare il programma introducendo direttamente i numeri decimali e rinunciando alla visualizzazione sullo schermo. Il programma potrebbe allora diventare:

```
  10 FOR x=1 TO 4
  20 READ n
  30 POKE (40000+2*x),n
  40 READ n
  50 POKE (40001+2*x),n
  60 NEXT x
  70 DATA BIN 10000100,200,BIN 10001000,200,BIN
11010000,200,BIN 00110000,200
  80 FOR x=1 TO 4
  90 OUT 31,PEEK (40000+2*x)
 100 PAUSE PEEK (40001+2*x)
 110 NEXT x
 120 GO TO 80
1000 FOR x=1 TO 10
1001 PRINT PEEK (40000+x)
1002 NEXT x
```

Questo programma opererà automaticamente, ma senza fornire all'utente alcuna possibilità di controllo. Una tipica sequenza di luci per albero di Natale o discoteca potrebbe essere:

Evento	*Intervallo*	*Luci*							
		1	2	3	4	5	6	7	8
1	100	1	0	0	0	0	0	0	0
2	80	0	1	0	0	0	0	0	0
3	60	0	0	1	0	0	0	0	0
4	40	0	0	0	1	0	0	0	0
5	20	0	0	0	0	1	0	0	0
6	40	0	0	0	0	0	1	0	0
7	60	0	0	0	0	0	0	1	0
8	80	0	0	0	0	0	0	0	1

L'effetto ottenuto è quello di luci che si rincorrono; la sequenza è ripetibile all'infinito rimpiazzando la linea 380 con un GOTO 300. È possibile preprogrammare varie sequenze da eseguire poi attraverso una serie di cicli FOR NEXT. Disponendo di circa 30000 locazioni di memoria dovrebbe essere possibile ottenere sequenze e configurazioni originali e complesse.

4.8 La subroutine "timer"

Il comando PAUSE dello Spectrum può essere usato per ottenere ritardi fino ad un massimo di 65535 intervalli, pari a circa 22 minuti. Il sequenziatore di eventi, che chiameremo in seguito *sequencer*, può comunque trattare solamente 225/50 di secondo, poiché una singola locazione di memoria può contenere numeri fino ad un massimo di 255.
È possibile programmare un timer nella routine sequencer presentata nel paragrafo 4.7 in modo da ottenere qualunque intervallo da pochi secondi fino a molte ore. Il programma o subroutine assume che Q sia l'intervallo da misurare. Q è immagazzinato nella locazione (40001+2*b) come numero compreso tra 0 e 255. La subroutine è comunque in grado di convertire Q in secondi, minuti, ore o anche giorni ed è contenuta nelle righe 600-630; ecco di seguito il listato del programma completo:

```
  10 DIM a(8): DIM b(100): DIM p(100): DIM q(
10)
  20 LET z=0: LET y=0: LET g=0
 100 INPUT "numero degli eventi  ";a: LET g=a
 105 PRINT g
 110 PRINT "evento          periodo"
 120 FOR n=1 TO a
 130 INPUT "batti un numero ad 8 bit        "
;b(n)
 140 PRINT n
 150 INPUT "batti l'intervallo (max 255)    "
;p(n)
 160 PRINT TAB 6;b(n);TAB 20;p(n)
 170 LET z=b(n)
 180 GO SUB 230
 190 POKE (40000+2*n),y
```

(continua)

```
 200 POKE (40001+2*n),p(n)
 210 NEXT n
 220 PRINT "premi un tasto per iniziare": PAU
SE 0: GO TO 300
 230 LET y=0
 235 FOR v=8 TO 1 STEP -1
 240 IF INT ((10^(v-1))-z)<=0 THEN  LET q(v)=
2^(v-1): GO TO 260
 250 IF INT ((10^(v-1))-z)>0 THEN  LET q(v)=0
: GO TO 270
 260 LET z=(z-(10^(v-1)))
 270 LET y=y+q(v)
 280 NEXT v
 290 RETURN
 300 FOR b=1 TO g
 310 LET y=PEEK (40000+2*b)
 320 OUT 31,y
 330 GO SUB 500
 340 PRINT "evento ";b;"  ";z
 350 PRINT TAB 17;"periodo ";b;"   ";PEEK (40
001+2*b)
 360 LET Q=PEEK (40001+2*b): GO SUB 600
 370 NEXT b
 380 STOP
 500 LET z=0
 505 FOR w=8 TO 1 STEP -1
 510 IF INT (y-(2^(w-1)))<0 THEN  LET a(w)=0:
 GO TO 540
 520 IF INT (y-(2^(w-1)))>=0 THEN  LET a(w)=1
0^(w-1)
 530 LET y=(y-(2^(w-1)))
 540 LET z=z+a(w)
 550 NEXT w
 560 RETURN
 600 POKE 23674,0: POKE 23673,0: POKE 23672,0
 610 LET t=INT ((65536*PEEK 23674+256*PEEK 23
673+PEEK 23672)/50)
 620 IF Q>=t THEN  RETURN
 630 GO TO 610
```

Nella linea 620 deve necessariamente comparire >= in quanto l'esatto momento in cui si avvera la condizione = potrebbe essere perso durante lo svolgimento del loop. La linea 360 del programma sequencer dovrebbe essere sostituita da:

```
360 LET q=PEEK (40001+2*b): GOSUB 600
```

Questa linea assume che q sia misurato in secondi. Se si desidera misurare in minuti, allora la linea 620 deve diventare:

```
IF q>=t*60 THEN RETURN.
```

Se invece si desidera misurare in ore, la linea 620 deve diventare:

```
IF q>=t*3600 THEN RETURN.
```

4.9 Mini consolle per effetti di luce in teatro

Gli effetti luminosi di qualunque spettacolo teatrale possono essere molto complessi. Il tecnico luci deve accendere e spegnere un certo numero di banchi di proiettori, focalizzare e dirigerne altri e nel contempo prestare attenzione al copione e all'azione che avviene in scena. Un esempio di illuminazione teatrale è mostrato in figura 4.10.
Il circuito di latch o il DAC più latch e basetta optotiristore può essere programmato per eseguire le sequenze di un programma di illuminazione per un gruppo di 8 luci o banchi di luci, così da lasciare al tecnico luci il solo compito di premere un qualunque tasto dello Spectrum per far avanzare la sequenza. In alternativa potrebbe essere possibile avanzare o ritardare la sequenza operando su certi altri tasti.
Il programma sequencer sarà pertanto modificato come segue:

```
linea 150 LET p (n)=0
linea 360 PAUSE 0
```

Così facendo la pressione di qualunque tasto farà avanzare la sequenza. Se si desidera anche la possibilità di ritardarla, la linea 360 sarà rimpiazzata da:

```
360 LET a$=INKEY$: IF a$=" " THEN GOTO 360:
                IF a$="R" THEN LET b=b-1: GOTO 310
```

Così il programma sarà anche in grado di procedere in direzione inversa

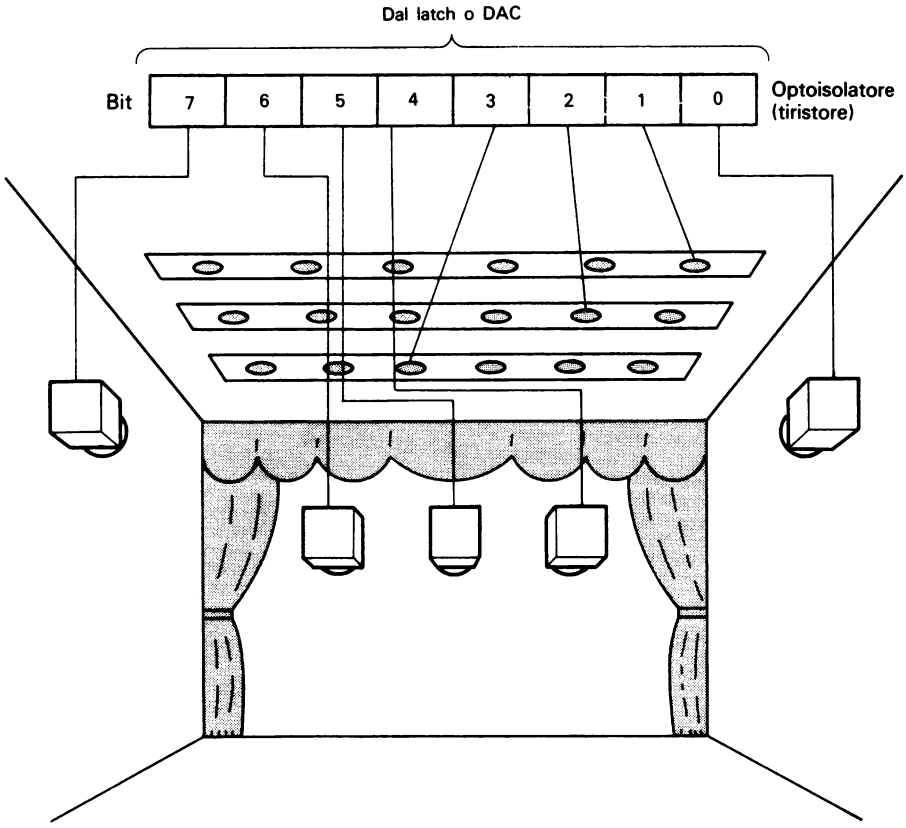


Figura 4.10 Esempio, semplificato al massimo, di disposizione di corpi illuminanti per l'illuminazione di un palcoscenico.

premendo il tasto R e in direzione normale premendo qualunque altro tasto.

4.10 Sistema di riscaldamento domestico

Un'utile applicazione sfrutta il sequencer più la subroutine "timer" programmata in intervalli di quarti d'ora (IF q>=t*750 in linea 620). Un tipico sistema di controllo per il riscaldamento domestico dispone solamente di due periodi di acceso/spento nelle 24 ore. Lo Spectrum metterà a disposizione tanti periodi d'accensione e di spegnimento quanti saranno richiesti, e questi periodi potranno anche variare in funzione del gior-

no della settimana oppure della settimana dell'anno. Inoltre sarà possibile controllare fino a 8 dispositivi esterni (ad esempio pompe). Molti sistemi di riscaldamento domestico sono estremamente costosi poiché riscaldano tutta la casa alla temperatura dell'ingresso mentre basterebbe distribuire il riscaldamento ogni ora in alcune stanze quando la casa è occupata. È possibile usare un optotiristore o optodiac per accendere o spegnere un massimo di otto dispositivi come pompe, allarmi anti-furto o anti-incendio, etc.
Quest'idea sarà sviluppata nel prossimo capitolo in cui useremo il convertitore analogico-digitale per misurare le temperature di vari punti e per controllare elettrovalvole tramite un dispositivo di latch.
Unico e ovvio svantaggio di questo sistema è che lo Spectrum non potrà essere usato per alcun altro scopo mentre si troverà collegato al sistema di riscaldamento.

4.11 Generatore di forme d'onda

Lo Spectrum è in grado di generare direttamente le funzioni SIN, COS, TAN, LN, EXP, e le funzioni circolari usando PI. È in grado di generare praticamente qualunque forma d'onda derivante da un'equazione fornita, anche se le forme d'onda più complesse richiederanno un certo tempo di calcolo e quindi la frequenza d'uscita sarà limitata.
Sfortunatamente, il tempo richiesto al computer per calcolare le varie funzioni restringe la frequenza d'uscita a solo pochi Hertz.
La forma d'onda di uscita può essere vista su uno schermo usando PLOT o DRAW e la corrispondente uscita analogica può essere misurata collegando un voltmetro all'uscita del DAC. Una sinusoide a lenta variazione, o altri segnali, possono essere usati per controllare direttamente la velocità di un motore o la posizione di un servocontrollo usando un programma come:

```
10 FOR n=0 TO 255
20 PLOT n, 88+80*SIN(n/128*PI)
30 OUT 31, 88+80*SIN(n/128*PI)
40 NEXT n
50 GOTO 10
```

Il disegno sarà più veloce omettendo dal listato la linea numero 30.
È più difficile ottenere uscite audio poiché la frequenza è bassa; la massima frequenza d'uscita ottenibile usando un programma BASIC è data da:

```
10 OUT 31, 0: OUT 31, 100
20 GOTO 10
```

È interessante rendersi conto del tempo richiesto dal computer per eseguire i conti e le altre varie operazioni, facendo girare il seguente programma:

```
 10 FOR x=1 TO 100
 20 OUT 31, x: OUT 31, 0
 30 NEXT x
 40 FOR x=1 TO 50
 50 OUT 31, 2*x: OUT 31, 0
 60 NEXT x
 70 FOR x=1 TO 1000, STEP 20
 80 OUT 31, x/20: OUT 31,0
 90 NEXT x
100 GOTO 10
```

Potremo udire tre segnali. Dovrebbero essere simili, ma, poiché la complessità del programma aumenta fra le linee 10-30, 40-60, e 70-90, il loro tono sarà differente. Maggiore è il numero di operazioni da eseguire, minore è la velocità di operazione e quindi il timbro di uscita.

Capitolo

Progetti ADC-DAC

5

I circuiti e le applicazioni descritte fino ad ora sfruttano o il convertitore analogico-digitale o il convertitore digitale-analogico. È possibile costruire alcuni progetti molto interessanti che sfruttano entrambi i circuiti, con segnali applicati agli ingressi dell'ADC nello stesso tempo in cui il latch o il DAC sta generando segnali di uscita. La memoria dello Spectrum viene qui sfruttata intensivamente per immagazzinare dati sia in maniera permanente, sia per brevi periodi.
Pertanto tutti i progetti presentati in questo capitolo richiedono:

1. Il convertitore analogico-digitale
2. Il convertitore digitale-analogico o la scheda latch
3. L'alimentatore per il convertitore digitale-analogico e la scheda latch
4. Gli appropriati trasduttori, ad esempio microfoni, amplificatori, termistori, interruttori, ecc.

Nella figura 5.1 possiamo vedere una scheda latch connessa ad una basetta DAC.

5.1 Progetti di registrazione e riproduzione del suono

Perché utilizzare lo Spectrum per registrare e riprodurre suoni quando un normalissimo registratore a cassette potrebbe andare altrettanto bene? Il segnale sonoro immagazzinato nella memoria dello Spectrum può

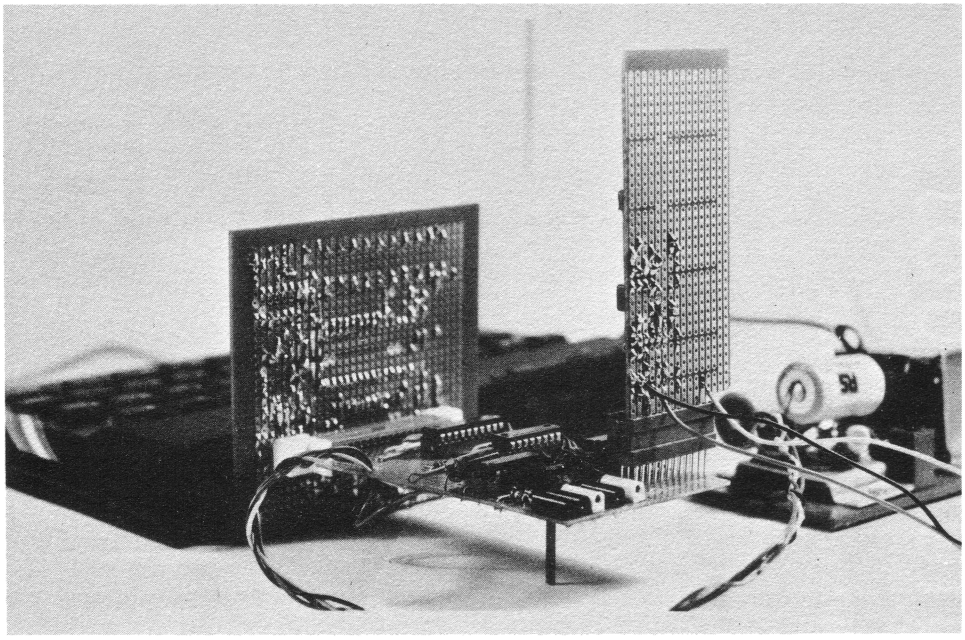


Figura 5.1 Inserimento della scheda latch nella basetta DAC.

essere manipolato accelerandolo, rallentandolo, riproducendolo al contrario, livellandolo, eseguendo montaggi e tante altre cose senza alcun bisogno né di un motore a velocità variabile né di una giuntatrice. Questo progetto registra un segnale fornito da un microfono, che potrà poi essere riprodotto in svariati modi per mezzo di semplici routine BASIC. Nel paragrafo 5.3 questa tecnica è usata per introdurre nello Spectrum fonemi rappresentanti le lettere dell'alfabeto uniti insieme in funzione di una parola digitata dall'utente. Lo Spectrum sembra parlare. È possibile registrare su nastro i fonemi per futuri usi. L'ultimo progetto sonoro del paragrafo 5.4 estende l'uso di questa tecnica alla realizzazione di una macchina generatrice di eco.
È importante notare che il computer sfrutta il campionamento per sintetizzare i suoni e questo spesso abbassa la qualità e la definizione del suono prodotto. La riproduzione fedele di un segnale a frequenza di 10 kHz della durata di 1 secondo richiederebbe 10000 celle di memoria; questo è ad esempio provato nel programma del paragrafo 5.2. Per poter registrare suoni della durata di vari secondi è necessario ridurre la frequenza di campionamento, e quindi anche la qualità. Si udirà pertanto il rumore di quantizzazione che darà al suono il caratteristico timbro da computer parlante.

5.2 Lo Spectrum usato come registratore

La procedura per registrare un suono in memoria è illustrata nella figura 5.2. Il programma di gestione è il seguente:

Codifica	*Decimale*	*Risultato*
ld de,40960	17	indirizzo iniziale del buffer
	0	per i campioni
	160	
push af	245	
push hl	229	
ld hl,5120	33	numero di campioni
	0	
	14	
in a,(30)	219	pone nell'accumulatore il segnale
	30	in ingresso
ld(de),a	18	e lo salva nelle locazioni puntate da DE
ld b,50	6	
	50	
dec b	5	ritardo per rallentare la frequenza
jr nz,−3	32	di campionamento (al posto di 50 può
	253	esserci un numero tra 0 e 255)
inc de	19	prepara il campionamento successivo
dec hl	43	
ld a,l	125	
or a	183	
jr nz,−9	32	contatore per 5120 campioni
	297	
ld a,h	124	
or a	183	
jr nz,−13	32	
	243	
pop hl	225	
pop af	241	
ret	201	ritorna al BASIC

La memoria dello Spectrum conterrà ora una successione di numeri compresi tra 0 e 255 in funzione del segnale fornito dal microfono. L'amplificatore operazionale presente sulla basetta ADC viene qui usato per amplificare il segnale fornito dal microfono. Le locazioni comprese tra 40960 e 46079 immagazzineranno il segnale sonoro digitalizzato per usi futuri.
L'operazione più semplice da compiere è ora riprodurre il suono registrato attraverso la basetta DAC. La procedura illustrata in figura 5.3, viene gestita dal seguente programma in codice-macchina:

Codifica	*Decimale*	*Risultato*
ld de, 40960	17	indirizzo iniziale del buffer contenente
	0	i campioni del suono digitalizzato
	160	
push af	245	
push hl	229	
ld hl,5120	33	numero di campioni
	0	
	14	
ld a,(de)	26	pone un campione nell'accumulatore
out (31),a	211	lo emette alla porta 31
	31	(il DAC)
ld b,50	6	
	50	ciclo di ritardo (50 in questo esempio)
dec b	5	
jr nz,−3	32	
	253	
inc de	19	prepara il campione successivo
dec hl	43	
ld a,l	125	
or a	183	
jr nz,−9	32	
	247	contatore per 5120 campioni
ld a,h	124	
or a	183	
jr nz,−13	32	

Codifica	*Decimale*	*Risultato*
	243	
pop hl	225	
pop af	241	
ret	201	ritorna al BASIC

Presentiamo ora un tipico programma BASIC che chiama le due routine in codice-macchina appena presentate e caricate una di seguito all'altra nelle locazioni comprese fra 64000 e 64057:

```
  10 FOR x=0 TO 57
  20 READ n
  30 POKE (64000+x),n
  40 NEXT x
  50 DATA 17,0,160,245,229,33,0,50,219,30,18,
6,50,5,32,253,19,43,125,183,32,242,124,183,32
,238,225,241,201
  55 DATA 17,0,160,245,229,33,0,50,26,211,31,
6,30,5,32,253,19,43,125,183,32,242,124,183,32
,238,225,241,201
  60 PRINT AT 0,0;"parla     ": PAUSE 0
  65 PRINT AT 0,7;">"
  70 LET l=USR 64000
  80 PRINT AT 0,0;"replay    ": PAUSE 0
  85 PRINT AT 0,7;">"
  90 LET l=USR 64029
 100 GO TO 60
```

Le linee comprese tra la 10 e la 55 provvedono a caricare il programma in codice-macchina. Le altre linee permettono di registrare il suono che sarà pronunciato dopo aver premuto qualunque tasto in risposta alla scritta "parla" che apparirà sullo schermo. Premendo qualunque tasto dopo l'apparizione della scritta "replay" potremo udire daccapo il suono che avevamo pronunciato servendoci di un amplificatore collegato all'uscita del DAC. Variando i tempi di ritardo di campionamento potremo variare il tono dei suoni. Potreste, ad esempio, provare a cambiare il valore 50 nella routine in codice-macchina di riproduzione per alzare o abbassare il tono.

5.3 Lo Spectrum parlante

Sono attualmente disponibili circuiti integrati in grado di "pronunciare" i nomi dei giorni e dei mesi dell'anno, termini tecnici, istruzioni ecc. Questi circuiti sono purtroppo costosi e duplicano una buona parte dell'elettronica già presente nello Spectrum. A noi basteranno le basette ADC-DAC e un microfono. Il seguente programma, che usa una buona parte della memoria dello Spectrum, immagazzina i suoni fonetici dell'alfabeto per mezzo di un microfono. La voce dell'utente crea le lettere dell'alfabeto. Digitando parole sulla tastiera dello Spectrum, il computer provvederà ad unire insieme i suoni fonetici per creare la parola richiesta. Il programma è estremamente semplice:

```
  10 FOR x=0 TO 57
  20 READ n
  30 POKE (64000+x),n
  40 NEXT x
  50 DATA 17,0,160,245,229,33,0,3,219,30,18,6
,100,5,32,253,19,43,125,183,32,242,124,183,32
,238,225,241,201
  55 DATA 17,0,160,245,229,33,0,3,26,211,31,6
,100,5,32,253,19,43,125,183,32,242,124,183,32
,238,225,241,201
  60 FOR x=0 TO 25
  70 POKE 64002,160+3*x
  80 PRINT "premi il tasto ";CHR$ (97+x);" e
parla": PAUSE 0
  90 LET l=USR 64000
 100 NEXT x
 110 INPUT "batti la parola";a$
 130 FOR x=1 TO LEN a$
 140 POKE 64031,160+(CODE a$(x TO x)-97)
 150 LET l=USR 64029
 160 NEXT x
 170 GO TO 110
```

La procedura per ottenere lo Spectrum parlante è illustrata in figura 5.4. È necessario pronunciare ogni singola lettera dell'alfabeto premendo il corrispondente tasto dello Spectrum. Quest'operazione sarà completata in pochi minuti. Il seguito è semplice: lo Spectrum dovrebbe parlare non

appena digitate le parole richieste. Alcune lettere hanno varie pronuncie. Il problema potrebbe essere risolto sfruttando una lista di fonemi più lunga ma, tenendo conto che ogni fonema occupa 768 locazioni, per le ventisei lettere dell'alfabeto saranno necessari circa 20K di memoria. La zona di memoria dedicata ai fonemi, cioè le locazioni da 40960 a 60928, può essere registrata su nastro. Per fare ciò occorre fermare il programma premendo i tasti SHIFT-BREAK dopo aver introdotto l'ultimo suono (quello corrispondente alla z), e digitando:

```
SAVE "parla" CODE 40960, 20000
```

È possibile ricaricare quanto salvato digitando:

```
LOAD "parla" CODE 40960, 20000
```

Ognuna di queste operazioni richiederà circa 90 secondi.
Una volta salvato e di nuovo ricaricato il codice, digitate la linea:

```
56 GOTO 110
```

in caso contrario, i fonemi saranno cancellati durante l'esecuzione della prima parte del programma.

5.4 Esperimenti di robotica controllati dallo Spectrum

Non è possibile definire completo un libro di computer che non faccia almeno riferimento ai robot. Il programma presentato in questo paragrafo farà sì che lo Spectrum riconosca un certo numero di comandi forniti per mezzo di un microfono e obbedisca facendo muovere una macchina in una direzione scelta. La macchina potrà essere ad esempio il braccio di un robot o congegni similari dotati di piccoli motori o servocomandi per realizzare il movimento (vedi Fig. 5.5). In figura 5.6a possiamo vedere un tipico braccio di robot in grado di muoversi rispondendo a quattro comandi di base. I motori si muovono singolarmente in risposta agli 8 bit del segnale di uscita dal circuito latch:

```
00000010 comando 1. indirizzo iniziale 40960
00000100 comando 2. indirizzo iniziale 41472
00001000 comando 3. indirizzo iniziale 41984
00010000 comando 4. indirizzo iniziale 42496
```

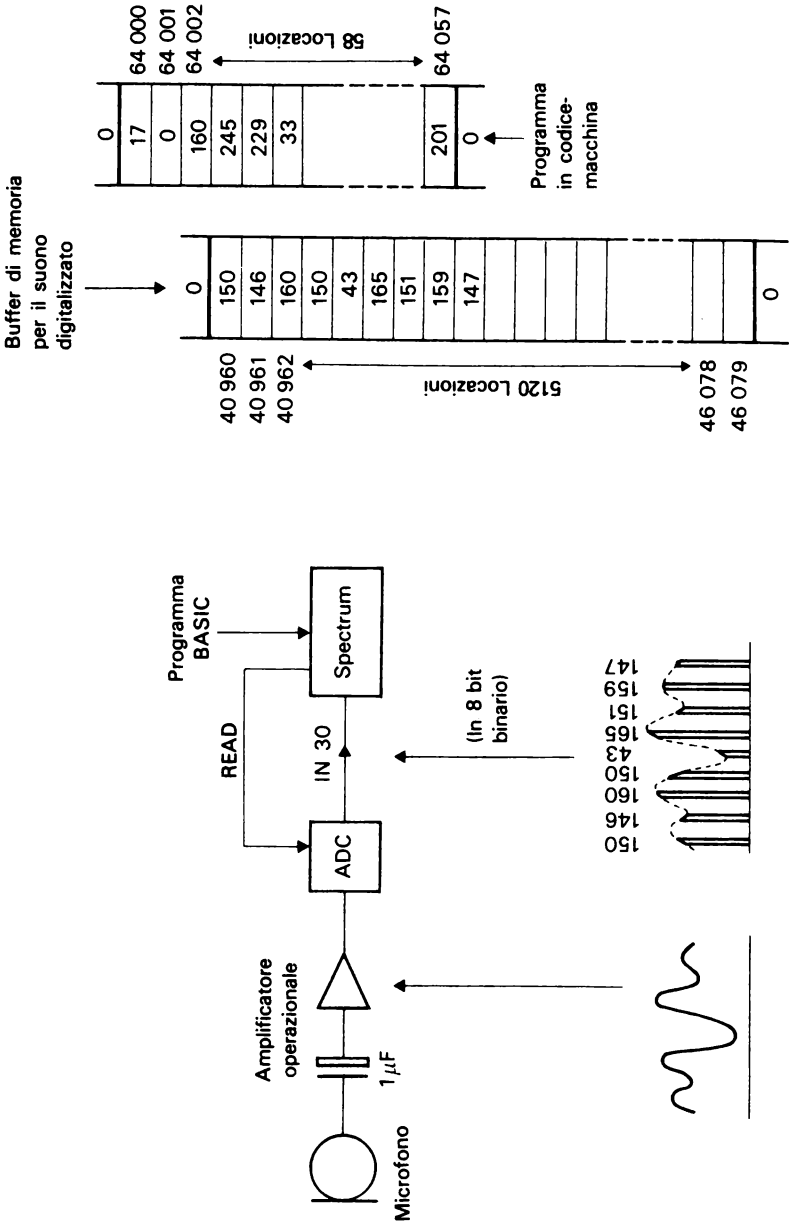


Figura 5.2 Uso dello Spectrum come registratore (voce 1).

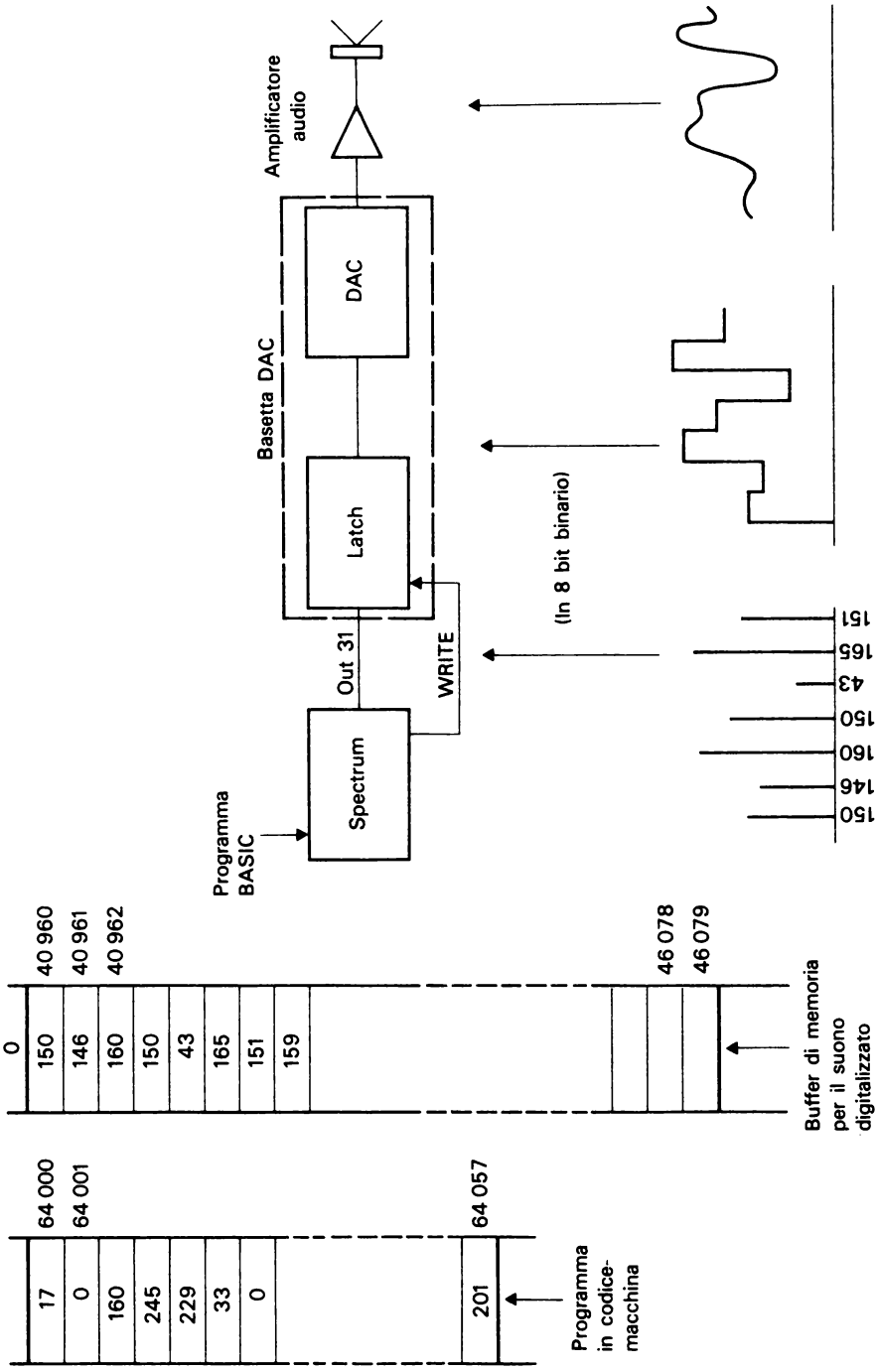


Figura 5.3 Uso dello Spectrum come riproduttore (voce 2).

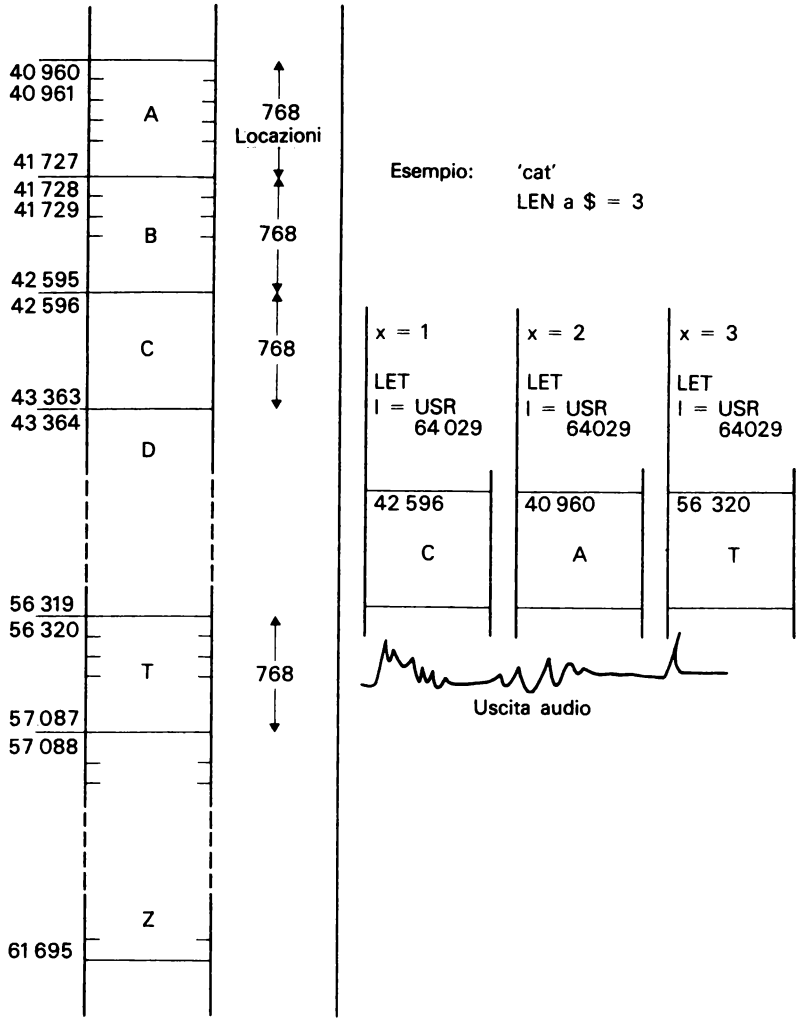


Figura 5.4 Lo Spectrum parlante.

Il programma accetta in sequenza quattro parole che immagazzina nelle locazioni di memoria sopra elencate. Ogni parola viene campionata 512 volte e, una volta memorizzata, resterà presente in memoria fino allo spegnimento dello Spectrum. Durante il processo di registrazione, ogni parola viene analizzata semplicemente sommando i dati ricavati dalla digitalizzazione. È pertanto consigliato scegliere come comandi parole di differente lunghezza e volume: parole come STOP, SU, GIÙ, hanno inviluppi simili mentre SOLLEVA e STOP hanno suoni molto diversi.

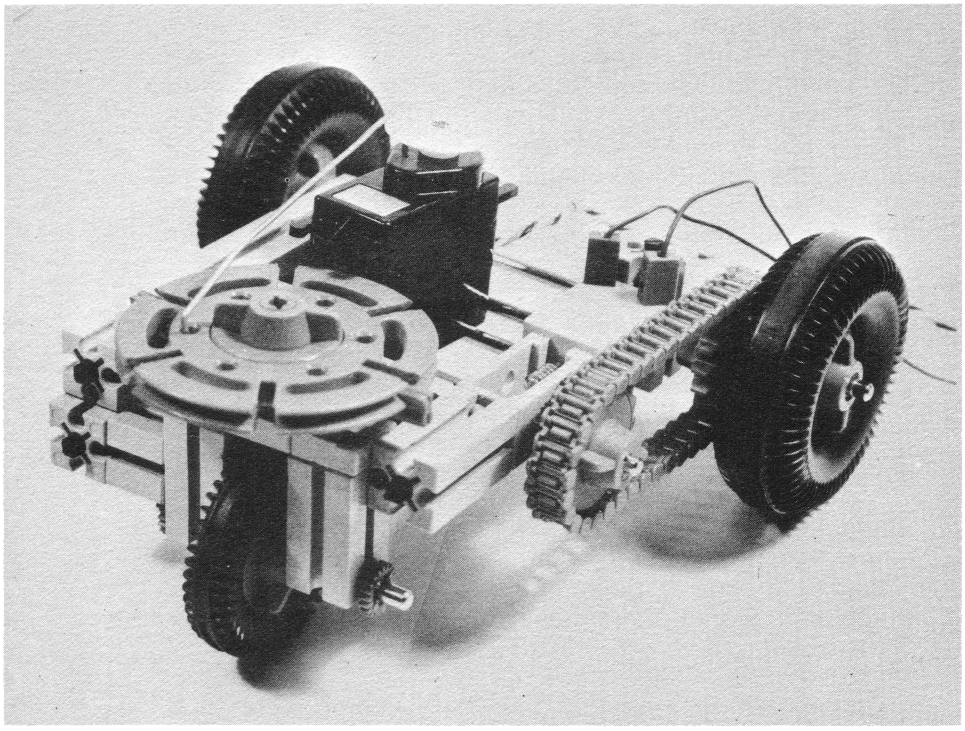


Figura 5.5 Piccolo robot mobile azionato da un motore principale e da un servocontrollo sterzante (ACOMS AS3) che sfrutta il programma generatore di impulsi.

Quando un comando viene ricevuto, viene immagazzinato in 512 locazioni di memoria inizianti in 43008 e analizzato. Il valore così ottenuto è quindi confrontato con quelli ricavati dai 4 comandi (immagazzinati in Z(1)-Z(4)) e un 1 logico viene messo al posto opportuno nel circuito latch corrispondente ad OUT 31.
Il comando deve essere pronunciato entro 1 secondo circa. Il tempo è limitato dal massimo numero inseribile nei registri dello Z80 e dal tempo richiesto per analizzare ogni parola. Indicatori di registrazione/calcolo comunicano all'utente l'operazione in corso.
La linea 400 prevede un ciclo di ritardo che inizia la registrazione quando l'utente parla e pertanto minimizza i periodi di silenzio.

```
   3 DIM z(10)
   5 PRINT "Pronuncia le seguenti parole": PR
INT : PRINT "in sequenza:": PRINT : PRINT "SU
": PRINT "GIU'": PRINT "SINISTRA": PRINT "DES
TRA": PRINT : PRINT "premi un tasto": PAUSE 0
  10 FOR x=0 TO 28
  20 READ n
  30 POKE (64000+x),n
  40 NEXT x
  50 DATA 17,0,160,245,229,33,0,2,219,30,18,6
,250,5,32,253,19,43,125,183,32,242,124,183,32
,238,225,241,201
  60 FOR c=0 TO 3
  70 POKE 64002,160+2*c
  80 CLS : PRINT "pronuncia ": GO SUB 301+c:
PRINT "premi un tasto": PAUSE 0
  85 PRINT AT 5,5;"registrazione"
  90 LET l=USR 64000
  95 PRINT AT 5,5;"elaborazione  "
 100 FOR p=0 TO 500
 110 LET z(c+1)=z(c+1)+PEEK (40960+(512*c)+p)
 120 NEXT p
 130 NEXT c
 140 PRINT "pronuncia il tuo comando": GO SUB
 400
 145 LET z(5)=0
 150 POKE 64002,168
 155 CLS : PRINT AT 0,8;"registrazione"
 160 LET l=USR 64000
 165 PRINT AT 0,8;"elaborazione"
 170 FOR p=0 TO 500
 180 LET z(5)=z(5)+PEEK (43008+p)
 190 NEXT p
 195 FOR d=1 TO 4
 200 IF z(5)-z(d)<1000 THEN  OUT 31,2^d: CLS
: GO SUB (300+d): OUT 31,0: GO TO 140
 210 NEXT d
 220 PRINT "Parola sconosciuta,riprova.": GO
TO 140
 301 PRINT AT 0,10;"SU        ": RETURN
 302 PRINT AT 0,10;"GIU'      ": RETURN
 303 PRINT AT 0,10;"SINISTRA " : RETURN
 304 PRINT AT 0,10;"DESTRA   "  : RETURN
 400 IF IN 30=0 THEN  GO TO 400
 410 RETURN
```

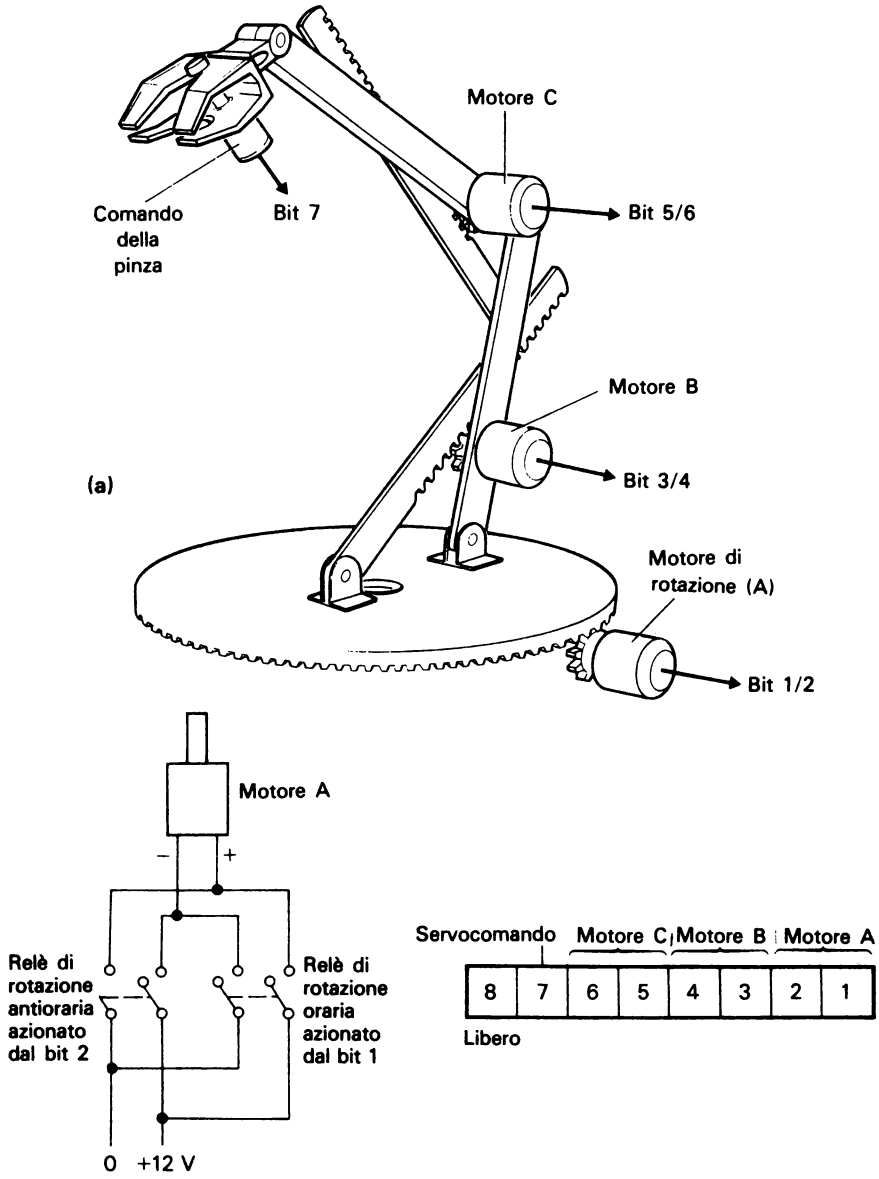


Figura 5.6 I programmi "robot". (a) Un tipico arto meccanico con il circuito latch controlla l'azionamento di tre motori e di un servocomando (pinza) per mezzo di una parola da 8 bit.

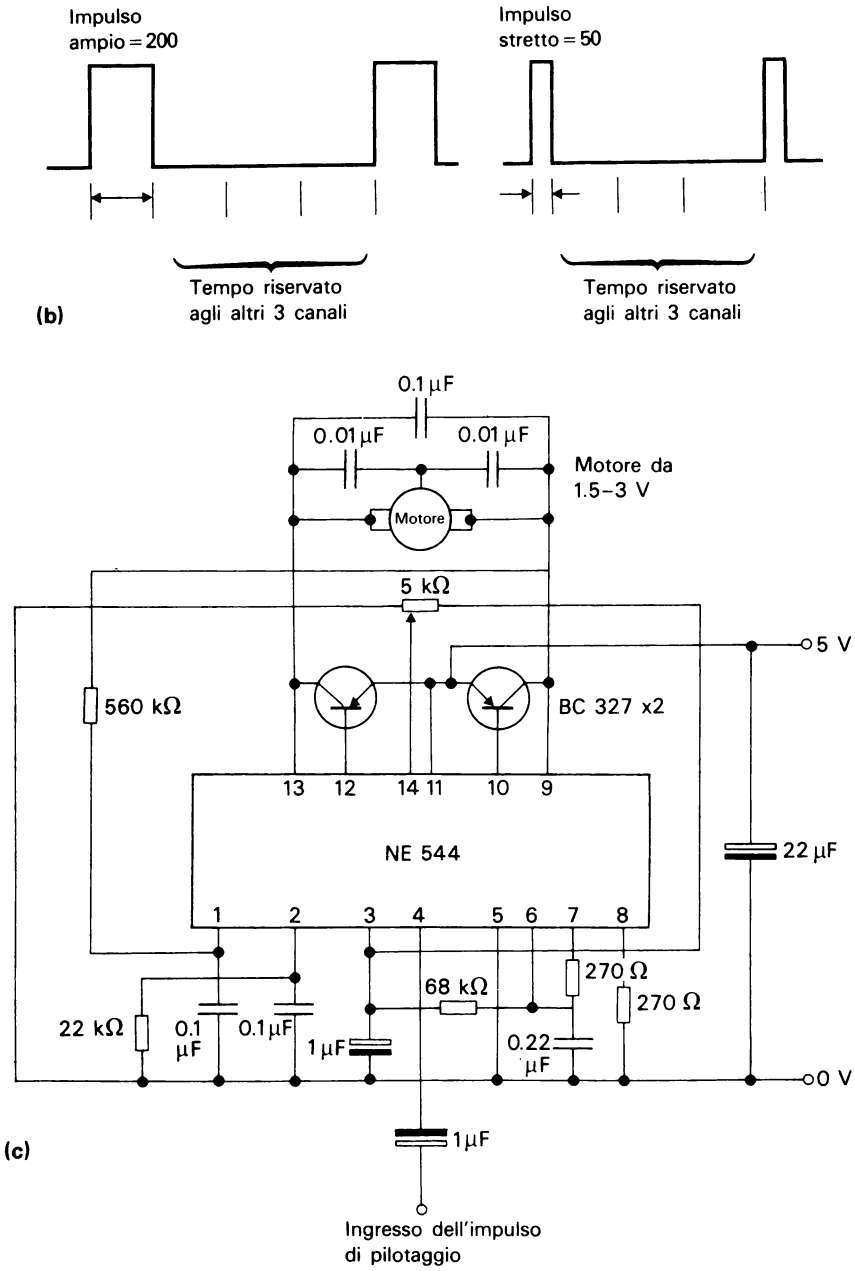


Figura 5.6 (continua). I programmi "robot": (b) Generazione degli impulsi per un canale in un sistema a quattro canali. (c) Schema elettrico di un servocontrollo elettronico.

Se la parola appare di difficile riconoscimento, si potranno usare serie di lunghezza variabile di BEEP come: BEEP, BEEP BEEP, BEEP BEEP BEEP, e così via.
Analizzatori di parola più sofisticati possono, ovviamente, riconoscere centinaia di parole, ma questo non è possibile con uno Spectrum da 48K. Questo progetto serve a dimostrare i principi di base.
Il prossimo programma "robot" usa il codice-macchina per controllare un robot ad otto canali per mezzo di una serie di joystick. Il programma si compone di alcuni generatori di impulsi, una serie per canale:

Codifica	*Decimale*	*Risultato*
in a,(24)	219	legge il joystick 1 e ne salva il valore in B
	24	
ld b,a	71	
ld a,1	62	identifica il canale 1 e inizia l'emissione dell'impulso
	1	
out (31),a	211	
	31	
dec b	5	stabilisce la durata dell'impulso 1
jr nz,−3	32	
ld a,0	253	
	0	
out (31),a	211	termina l'impulso
	31	

Se si desidera usare due canali così concepiti, un completo programma in codice-macchina è:

Codifica	*Decimale*	*Risultato*
in a,(24)	219	locazione 64000
	24	
ld b,a	71	
ld a,1	62	
	1	
out (31),a	211	
	31	canale 1
dec b	5	
jr nz,−3	32	
	253	
	62	
ld a,0	0	
out (31),a	211	
	31	

in a,(25)	219	
	25	
ld b,a	71	
ld a,2	62	
	2	
out (31),a	211	
	31	canale 2
dec b	5	
jr nz,−3	32	
	253	
ld a,0	62	
	0	
out (31),a	211	
	31	

Codifica	*Decimale*	*Risultato*
ld b,255	6	
	255	
dec b	5	
jr nz,−3	32	due cicli di ritardo necessari
	253	a distanziare gli impulsi quando
ld b,255	6	si usano meno di cinque canali
	255	
dec b	5	
jr nz,−3	32	
	253	
jp 64000	195	
	255	salto all'inizio della routine
	249	

Gli impulsi di uscita appaiono, come in figura 5.6b, di ampiezza variabile fra 0 e 2 millisecondi per canale; gli impulsi sono spaziati fra loro di 15-20ms. Questo è il tipo di segnale richiesto per pilotare un servocontrollo come il modello ACOMS AS-3, che è un servocomando universale sfruttato in un vasto campo di sistemi di radiocontrollo. L'unità è completamente autocontenuta, molto potente e può essere pilotata direttamente dal circuito di latch e dalla routine in codice-macchina sopra riportata. Il servocontrollo contiene un sistema di retroazione integrale che permette posizionamenti di precisione con il potenziometro di retroazione accoppiato direttamente alla ruota di uscita. Il circuito di un simile servocontrollo è presentato in figura 5.6c ed è destinato a coloro che possono disporre di questi componenti e sono in grado di costruire questi circuiti per controllare i movimenti di robot. Nei modelli radiocontrollati è necessario usare componenti in miniatura così da avere unità estremamente compatte, ma, per apparecchiature come quella presentata, destinata a stare su un pavimento e di natura fondamentalmente sperimentale, dimensioni e peso non sono particolarmente importanti. I servocontrolli comunque sono abbastanza economici per essere comprati già pronti.
Si suppone che i collegamenti tra lo Spectrum e il robot siano effettuati per mezzo di fili. Se si desidera trasmettere il segnale di controllo via radio, impulsi sonori, o raggi di luce, saranno necessarie alcune piccole modifiche al programma e, in particolare, occorrerà caricare il registro A sempre con 1, anziché con 1, 2, 4, 8, e così via per ogni canale. Il segnale

di uscita dal canale 1 sarà ora una successione di impulsi di ampiezza variabile, che saranno poi decodificati nel robot da un ricevitore di controllo standard. Viene lasciata all'utente la determinazione dei collegamenti meccanici da effettuare col modello. L'alimentazione in continua da fornire ai servocontrolli e ai motori dovrà provenire da una sorgente esterna che potrà essere costituita da batterie da 4.5 V di tipo classico, batterie al nickel-cadmio da 4.8 V, o da un alimentatore stabilizzato a 5 V analogo a quello usato per la basetta DAC.
Nella mia realizzazione, ho prelevato appunto dalla basetta DAC la tensione necessaria ad alimentare il robot. Il programma di gestione dipende dal numero di canali. Il segnale per ogni canale viene deviato all'opportuna uscita latch inserendo 1, 2, 4, 8 o 16 nel comando OUT 31.
Per due canali digitate:

```
10 FOR x=0 to 40
20 READ n
30 POKE (64000+x), n
40 NEXT x
50 DATA 219, 24, 71, 62, 1, 211, 31, 5, 32, 253, 62, 0, 211, 31, 219, 25,
   71, 62, 2, 211, 31, 5, 32, 253, 62, 0, 211, 31, 6, 255, 5, 32, 253, 6,
   255, 5, 32, 253, 195, 255, 249
60 LET 1=USR 64000
```

Salvate il programma e lanciatelo.
Un programma in BASIC che sfrutta l'ADC, un microfono con relativo amplificatore e la basetta latch (o DAC) risponde fino ad otto differenti comandi semplicemente pronunziando qualcosa nel microfono:

```
  10 PRINT "LE SEGUENTI PAROLE APPARIRANNO":
PRINT : PRINT "SULLO SCHERMO AD INTERVALLI":
PRINT : PRINT "DI UN SECONDO": PRINT
  20 INK 2: PRINT "SINISTRA": PRINT : PRINT "
DESTRA": PRINT : PRINT "SU": PRINT : PRINT "G
IU'"
  30 INK 0: PRINT : PRINT "PER ATTIVARE E ARR
ESTARE": PRINT : PRINT "IL CONTROLLO VOCALE,
": PRINT : PRINT "PREMI UN TASTO"
  40 PAUSE 0: CLS
  50 FOR x=1 TO 4: GO SUB 300+x
  60 FOR z=1 TO 100: IF IN 30=0 THEN  NEXT z:
 GO TO 80
```

(continua)

```
 70 OUT 31,2^x: CLS : PRINT AT 3,0;"ATTIVATO
": GO SUB 400: OUT 31,0: CLS
 80 NEXT x
 90 GO TO 50
301 PRINT AT 0,8;"SU        ": RETURN
302 PRINT AT 0,8;"GIU'      ": RETURN
303 PRINT AT 0,8;"SINISTRA ": RETURN
304 PRINT AT 0,8;"DESTRA   ": RETURN
400 IF IN 30=0 THEN  GO TO 400
410 RETURN
```

```
LE SEGUENTI PAROLE APPARIRANNO
SULLO SCHERMO AD INTERVALLI
DI UN SECONDO
SINISTRA
DESTRA
SU
GIU'
PER ATTIVARE E ARRESTARE
IL CONTROLLO VOCALE,
PREMI UN TASTO
```

I comandi appaiono sullo schermo in sequenza per un secondo ciascuno. Alla lettura del comando desiderato, l'utente genera un suono facendo così apparire alla corrispondente uscita della basetta latch un 1 logico. Questo segnale permarrà sull'uscita fino al successivo suono emesso dall'utente. La sequenza si può ripetere all'infinito. Desiderando, ad esempio, controllare i movimenti di una gru, l'utente dovrà generare un suono all'apparire sullo schermo della parola "su". Basterà generare un ulteriore suono quando il modellino avrà raggiunto la posizione desiderata. Il motore di controllo della salita sarà attivato da un 1 logico sulla corrispondente uscita latch per mezzo di un optoaccoppiatore e un relè o triac. Desiderando poi muovere il braccio a destra o a sinistra, occorrerà ripetere la sequenza.
Il circuito microfonico è progettato per rispondere a qualunque tipo di suono, quindi andrà bene anche uno schiocco di dita. IN 30 non sarà mai

0 se si opera in presenza di un forte rumore di sottofondo, quindi occorrerà alzare la soglia a valori intorno a 50. L'istruzione di linea 60 andrà pertanto modificata in: IF IN 30=50... È inoltre possibile far generare il BEEP al programma, ma questo è lasciato alla fantasia dell'utente.
È ovviamente possibile, per soddisfare necessità personali, alterare il numero e la forma dei comandi disponibili.

5.5 Una camera d'eco

Per mezzo di un semplice programma in codice-macchina, le capacità di registrazione del complesso ADC-DAC permettono allo Spectrum di operare come una camera d'eco automatica. Il programma è il seguente:

Codifica	*Decimale*	*Risultato*
ld de,40960	17	(64000 – indirizzo del programma)
	0	
	160	indirizzo iniziale del suono
ld hl,5120	33	
	0	
ld a,(de)	14	numero di campioni per ciclo
	26	
out(31),a	211	emissione ed acquisizione
	31	
in a,(30)	219	
	30	
ld(de),a	18	
ld b,50	6	ciclo di ritardo, controllato dal registro B
	50	
dec b	5	
jr nz, – 3	32	
	253	

Codifica	*Decimale*	*Risultato*
inc de	19	contatore
dec hl	43	
ld a,l	125	
or a	183	
jr nz,−7	32	
	239	
ld a,h	124	
or a	183	
jr nz,−21	32	
	235	
jp	195	ritorna all'inizio del ciclo
64005	255	
	249	

Il programma riempie le locazioni a partire da 40960 come il semplice programma di registrazione e riproduzione. Riempite le locazioni e, lasciato passare l'intervallo richiesto, il programma ritorna alla locazione 40960 ed emette il primo byte; simultaneamente la stessa locazione viene riprogrammata da un nuovo segnale microfonico. Questa sequenza si ripete ciclicamente con la riproduzione ritardata di un secondo o più rispetto alla registrazione, formando così un eco. La sequenza viene chiarificata nel seguente diagramma, in cui si suppone di aver appena lanciato il programma:

	1		2		3	
	OUT	IN	OUT	IN	OUT	IN
40960	0	a	a	aa	aa	aaa
40961	0	b	b	bb	bb	bbb
40962	0	c	c	cc	cc	ccc
40963	0	e	e	ee	ee	eee
'	0	f	f	ff	ff	fff
'	0	g	g	gg	gg	ggg
'	0	h	h	hh	hh	hhh
'	0	i	i	ii	ii	iii
'	0	j	j	jj	jj	'
'	0	k	k	kk	kk	'
'	0	l	l	ll	ll	'
		'		'		'

```
10 FOR x=0 TO 29
20 READ n
30 POKE (64000+x),n
40 NEXT x
50 DATA 17,0,160,33,0,14,26,211,31,219,30,1
8,6,50,5,32,253,19,43,125,183,32,239,124,183,
32,235,195,255,249
60 PRINT "ECO": PRINT : PRINT "Una volta in
iziata l'esecuzione": PRINT : PRINT "non e' p
ossibile arrestare": PRINT : PRINT "il progra
mma.": PRINT : PRINT "E' possibile variare il
 ritardo": PRINT : PRINT "bloccando il progra
mma e": PRINT : PRINT "cambiando il numero ch
e segue": PRINT : PRINT "il 6 in linea 50 con
 un altro": PRINT : PRINT "numero compreso tr
a 0 e 255.": PRINT : PRINT "PREMI UN TASTO"
70 PAUSE 0
80 LET l=USR 64000
```

```
 ECO
Una volta iniziata l'esecuzione
non e' possibile arrestare
il programma.
E' possibile variare il ritardo
bloccando il programma e
cambiando il numero che segue
il 6 in linea 50 con un altro
numero compreso tra 0 e 255
PREMI UN TASTO
```

Poiché questo è un programma in codice-macchina, una volta iniziato, non è possibile fermarne l'esecuzione: il programma continuerà a produrre echi fino allo spegnimento del computer.
Esistono due modi per regolare la lunghezza dell'eco; innanzitutto cam-

biando il numero (caricato nella coppia di registri HL) che determina la lunghezza della scansione; è inoltre possibile variare la lunghezza della routine di ritardo (contenuta nel registro B) modificando il numero che segue il 6 nella linea 50. Per ottenere suoni di alta qualità mantenete basso il numero da caricare in B e alto il numero da caricare in HL, anche se questo richiederà molti Kbyte di memoria.

5.6 Sistema di controllo per trenini elettrici

Questo progetto, che usa il blocco ADC-DAC, è in grado di controllare otto componenti di un plastico per trenini elettrici (sezioni di binari, luci, segnalazioni, ...).
I componenti del plastico sono controllati dalla basetta latch per mezzo di optoaccoppiatori e relè o di V-FET. Gli elementi di programma sono i seguenti:

1–6	Programma in codice-macchina che disegna il tracciato della ferrovia
7–9	Memorizzazione del tracciato dei binari del trenino
10–80	Istruzioni
90–210	Memorizzazione dei dati, come col sequencer di eventi
230–290	Subroutine di conversione da binario a decimale
300–360	Sequencer di eventi
365	Sensore di binario per controllare la posizione del treno
500–560	Subroutine di conversione da decimale a binario
610–690	Subroutine del tracciato dei binari

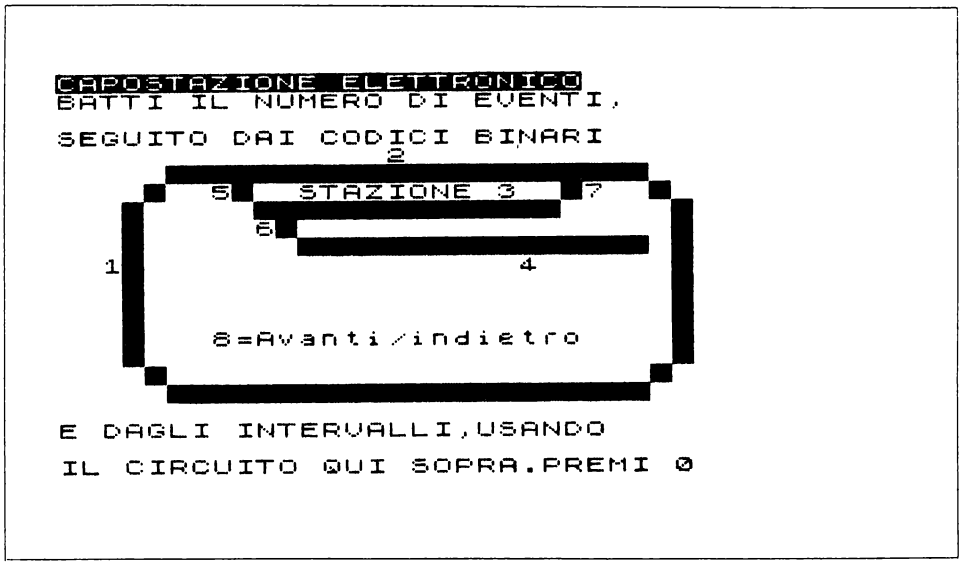
```
   1 BORDER 5: PAPER 5: INK 0: CLEAR 44500:
CLS
   2 FOR x=0 TO 41
   3 READ n
   4 POKE (65240+x),n
   5 NEXT x
   6 DATA 17,215,227,33,0,64,6,27,197,6,0,126
,18,19,35,16,249,193,16,244,201,17,0,64,33,21
5,227,6,27,197,6,0,126,18,19,35,16,249,193,16
,244,201
   7 FOR g=610 TO 690 STEP 10: GO SUB g: NEXT
 g
   9 LET l=USR 65240
```

(continua)

```
  10 DIM a(8): DIM b(100): DIM p(100): DIM q(
10): DIM c(8)
  20 LET z=0: LET y=0: LET g=0
  40 PRINT AT 0,0;"CAPOSTAZIONE ELETTRONICO"
  50 PRINT "BATTI IL NUMERO DI EVENTI, ": PRI
NT
  60 PRINT "SEGUITO DAI CODICI BINARI"
  70 PRINT AT 19,0;"E DAGLI INTERVALLI,USANDO
": PRINT
  80 PRINT "IL CIRCUITO QUI SOPRA.PREMI 0"
  90 PAUSE 0: LET l=USR 65261
 100 INPUT "numero di eventi   ";a: LET g=a
 110 PRINT "evento          periodo"
 120 FOR n=1 TO a
 130 INPUT "batti un numero ad 8 bit      ";b
(n)
 140 PRINT n
 150 INPUT "batti l'intervallo (max 255)   ";
p(n)
 160 PRINT TAB 6;b(n);TAB 20;p(n)
 170 LET z=b(n)
 180 GO SUB 230
 190 POKE (40000+2*n),y
 200 POKE (40001+2*n),p(n)
 210 NEXT n
 220 CLS : PRINT "PREMI UN TASTO PER INIZIARE
": PAUSE 0
 225 GO TO 300
 230 LET y=0
 235 FOR v=8 TO 1 STEP -1
 240 IF INT ((10^(v-1))-z)<=0 THEN  LET q(v)=
2^(v-1): GO TO 260
 250 IF INT ((10^(v-1))-z)>0 THEN  LET q(v)=0
: GO TO 270
 260 LET z=(z-(10^(v-1)))
 270 LET y=y+q(v)
 280 NEXT v
 290 RETURN
 300 FOR b=1 TO g
 305 LET l=USR 65261
 310 LET y=PEEK (40000+2*b)
 320 OUT 31,y
 330 GO SUB 500
 340 PRINT AT 0,0; INK 0;"evento ";b;"  ";z
```

(continua)

```
 350 PRINT TAB 17; INK 7;"periodo ";b;"   ";P
EEK (40001+2*b)
 360 PAUSE PEEK (40001+2*b)
 364 FOR w=1 TO 4
 365 IF IN (c(w))<200 THEN  GO TO 365
 366 NEXT w
 370 NEXT b
 380 LET l=USR 65261: PRINT "FINITO. PREMI UN
 TASTO PER RIPETERE.": PAUSE 0: GO TO 300
 500 LET z=0
 505 FOR w=8 TO 1 STEP -1
 510 IF INT (y-(2^(w-1)))<0 THEN  LET a(w)=0:
 GO TO 540
 520 IF INT (y-(2^(w-1)))>=0 THEN  LET a(w)=1
0^(w-1): FLASH 1: GO SUB 600+10*w: FLASH 0: L
ET c(w)=23+w
 530 LET y=(y-(2^(w-1)))
 540 LET z=z+a(w)
 550 NEXT w
 560 RETURN
 610 INK 0: PRINT AT 5,5;" ";AT 5,6;" ";AT 5,
25;" ";AT 5,26;" ": PRINT AT 16,4;" ";AT 16,2
7;" ";AT 6,4;" ";AT 6,27;" ": FOR x=1 TO 9: P
RINT AT (6+x),3;" ";AT (6+x),28;" ": NEXT x:
FOR x=1 TO 22: PRINT AT 17,(4+x);" ": NEXT x:
 RETURN
 620 FOR x=1 TO 12: PRINT AT 5,(9+x);" ": NEX
T x: RETURN
 630 INK 1: FOR x=12 TO 21: PRINT AT 7,x;" ":
 NEXT x: RETURN
 640 INK 6: FOR x=12 TO 26: PRINT AT 9,x;" ":
 NEXT x: RETURN
 650 INK 2: PRINT AT 5,7;" ";AT 6,8;" ": RETU
RN
 660 INK 4: PRINT AT 7,9;" ";AT 8,10;" ";AT 9
,11;" ": RETURN
 670 INK 2: PRINT AT 5,24;" ";AT 6,23;" ";AT
7,22;" ": RETURN
 680 PRINT AT 14,7; INK 0;"8=Avanti/indietro"
: RETURN
 690 INK 2: PRINT AT 5,8;" ";AT 5,9;" ";AT 5,
22;" ";AT 5,23;" ": INK 4: PRINT AT 7,10;" ";
AT 7,11;" ": INK 0: PRINT AT 4,15;"2";AT 6,11
;"STAZIONE";AT 6,20;"3";AT 6,24;"7";AT 6,7;"5
";AT 8,9;"6";AT 10,21;"4";AT 10,2;"1": RETURN
```



Il programma può controllare uno o due treni su un semplice circuito ad un binario, come quello di figura 5.7; i treni saranno gestiti lungo i punti e le sezioni di binario fermandosi quando necessario. Un treno può camminare finché non sia rilevato da un sensore che può essere tanto un rivelatore a fotocellula (composto da una luce e da un rivelatore di luce come quello descritto nel paragrafo 3.6) o una sezione di binario che venga cortocircuitata dalle ruote del treno. È possibile istruire opportunamente la linea 365, o, se si preferisce, ometterla.
È possibile anche il controllo manuale del movimento (in modo simile al programma sequencer) premendo qualunque tasto.
Questo si ottiene rimpiazzando la linea 360 con:

```
360 IF INKEY $=" " THEN GOTO 360
```

o

```
360 PAUSE 0
```

È possibile eseguire controlli antideragliamento, aggiungendo la seguente linea 185 che filtra i movimenti illeciti:

```
185 IF p(1)=p(5)=p(2)=1  OR
    (p(5)=1 AND p(7)=0)  OR
    p(2)=p(3)=1          OR
```

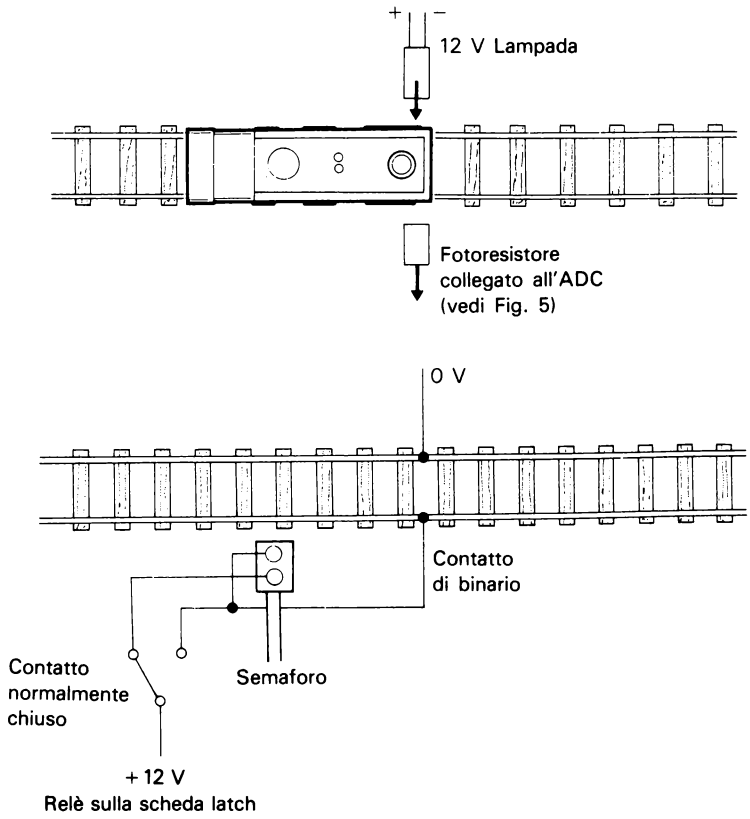


Figura 5.7 Il programma "treno". (a) Rilevamento della posizione del treno per disattivare l'alimentazione al binario. (b) Un semplice semaforo ferroviario connesso al relè di disattivazione binario sulla scheda latch.

```
p(3)=p(4)=1          OR
p(2)=p(7)=p(1)=1     OR
p(2)=p(4)=1
THEN PRINT "MOVIMENTO ILLECITO": GOTO 130
```

5.7 Gestione dell'energia domestica

Nel paragrafo 4.10 un sistema sequencer controllava direttamente, ad intervalli di 15 minuti, fino ad otto sistemi di pompe o valvole. Aggiungendo sensori di temperatura simili al termometro descritto nel paragrafo

3.10, lo Spectrum può controllare direttamente la temperatura in otto aree, provvedendo a comandare le necessarie pompe, valvole o ventilatori. Un tipico programma potrebbe essere:

```
110 DIM p (10)
120 LET z=0
130 FOR x=1 TO 8
140 LET p(x)=0
150 IF IN (23+x)<20 THEN LET p(x)=2 ↑ (x-1)
160 NEXT x
170 FOR x=1 TO 8
180 LET z=z+p(x)
190 NEXT x
200 OUT 31, z
210 GO TO 120
```

Le linee 130-160 fungono da rivelatori di bassa temperatura e piazzano un 1 binario nell'appropriato segnale di uscita al latch.
Le linee 170-190 calcolano il necessario numero decimale da emettere in forma binaria a 8 bit in linea 200.
Per esempio, se le stanze 2 e 6 sono fredde, p(2)=2 e p(6)=32, quindi z=34 sarà emesso al latch sotto forma di 001000100 per accendere le pompe 2 e 6.
Se si desidera controllare otto ambienti a temperature diverse, occorrerà programmare opportunamente (nelle linee da 130 a 160) i valori p(1)-p(8), per esempio con:

```
130 FOR x=1 TO 5
140 LET p(x)=0
150 IF IN 24<20 THEN LET p(1)=1:
    IF IN 25<25 THEN LET p(2)=2:
    IF IN 26<30 THEN LET p(3)=4:
    IF IN (26+x)<40 THEN LET p(3+x)=2 ↑ (2+x)
160 NEXT x
```

È possibile anche controllare temperature eccessive sfruttando sistemi di condizionamento d'aria o ventilatori, cambiando la linea 150 in modo che riconosca valori > e non <. La temperatura da misurare può variare dalla temperatura di congelamento con, ad esempio, un rivelatore di ghiaccio in un garage, fino al punto di ebollizione per riscontrare un malfunzionamento in uno scaldabagno. È possibile inserire nel programma allarmi sonori sfruttando, ad esempio:

```
IF IN 24<5 THEN LET p(1)=1:BEEP 1, 10
```

Questo programma, accoppiato con il sequencer di eventi, controllerà qualunque sistema di riscaldamento domestico usando il programma di rilevazione della temperatura come una subroutine:

```
 10 POKE 23674, 0: POKE 23673, 0: POKE 23672, 0
 20 LET t=INT((65536*PEEK 23674+256*PEEK 23673+
    PEEK 23672)/50)
 30 IF q>=7*t*3600
    OR q<=10*t*3600
    OR q>15*t*3600
    OR q<=22*t*3600
    THEN GO SUB 110
 40 IF q>=24*t*3600 GO TO 10
110-210 come il programma precedente
220 RETURN
```

Questo programma manterrà le temperature richieste durante gli orari compresi fra le 7 e le 10 e fra le 15 e le 22. È possibile inserire, se necessario, routine di controllo più complesse.

```
  10 DIM a(8): DIM b(100): DIM p(100): DIM q(
10)
  20 LET z=0: LET y=0: LET g=0
 100 INPUT "numero di eventi  ";a: LET g=a
 105 PRINT g
 110 PRINT "evento          periodo"
 120 FOR n=1 TO a
 130 INPUT "batti un numero ad 8 bit        "
;b(n)
 140 PRINT n
 150 INPUT "batti l'intervallo (max 255) ";p(
n)
 160 PRINT TAB 6;b(n);TAB 20;p(n)
 170 LET z=b(n)
 180 GO SUB 230
 190 POKE (40000+2*n),y
 200 POKE (40001+2*n),p(n)
 210 NEXT n
 220 PRINT "premi un tasto per iniziare": PAU
SE 0: GO TO 300
 230 LET y=0
 235 FOR v=8 TO 1 STEP -1
 240 IF INT ((10^(v-1))-z)<=0 THEN  LET q(v)=
```

(continua)

```
2^(v-1): GO TO 260
 250 IF INT ((10^(v-1))-z)>0 THEN  LET q(v)=0
: GO TO 270
 260 LET z=(z-(10^(v-1)))
 270 LET y=y+q(v)
 280 NEXT v
 290 RETURN
 300 FOR b=1 TO g
 310 LET y=PEEK (40000+2*b)
 320 OUT 31,y
 330 GO SUB 500
 340 PRINT "evento ";b;"  ";z
 350 PRINT TAB 17;"periodo ";b;"   ";PEEK (40
001+2*b)
 360 LET Q=PEEK (40001+2*b): GO SUB 600
 365 GO SUB 1110
 370 NEXT b
 380 GO TO 300
 500 LET z=0
 505 FOR w=8 TO 1 STEP -1
 510 IF INT (y-(2^(w-1)))<0 THEN  LET a(w)=0:
 GO TO 540
 520 IF INT (y-(2^(w-1)))>=0 THEN  LET a(w)=1
0^(w-1)
 530 LET y=(y-(2^(w-1)))
 540 LET z=z+a(w)
 550 NEXT w
 560 RETURN
 600 POKE 23674,0: POKE 23673,0: POKE 23672,0
 610 LET t=INT ((65536*PEEK 23674+256*PEEK 23
673+PEEK 23672)/50)
 620 IF Q>=t THEN  RETURN
 630 GO TO 610
1110 DIM p(10)
1120 LET z=0
1130 FOR x=1 TO 8
1140 LET p(x)=0
1150 IF IN (23+x)<20 THEN  LET p(x)=2^(x-1)
1160 NEXT x
1170 FOR x=1 TO 8
1180 LET z=z=p(x)
1190 NEXT x
1200 OUT 31,z
1210 GO TO 1120
1220 RETURN
```

```
evento          periodo
1
        1000            5
2
        100             5
3
        10              5
premi un tasto per iniziare
evento 1  1000
                  periodo 1    5
evento 2  100
                  periodo 2    5
evento 3  10
                  periodo 3    5
evento 1  1000
                  periodo 1    5
evento 2  100
                  periodo 2    5
evento 3  10
                  periodo 3    5
evento 1  1000
                  periodo 1    5
```

5.8 Sistema d'allarme antifurto

Questo programma usa gli otto ingressi della basetta ADC, insieme al DAC o alla basetta latch per generare un allarme ogniqualvolta uno degli ingressi assume valori superiori a 10.

```
  10 FOR x=1 TO 8
  20 PRINT AT 2*x,0;"ingresso ";x;"  ";IN (23
+x)
  30 IF IN (23+x)>10 THEN  GO TO 100
  40 NEXT x
  50 GO TO 10
 100 PAUSE 100
 110 IF IN (23+x)>10 THEN  GO TO 130
 120 GO TO 40
 130 CLS : FLASH 1: PRINT "effrazione all'ing
resso ";x;"        allarme"
 140 OUT 31,255: GO TO 140
```

```
ingresso 1  191
ingresso 2  191
ingresso 3  191
ingresso 4  191
ingresso 5  255
ingresso 6  255
ingresso 7  254
ingresso 8  255
```

Il programma fornisce, se necessario, un segnale sullo schermo televisivo. Gli ingressi possono essere:

1. Semplici pulsanti connessi alle porte e alle finestre, in cui il circuito chiuso corrisponda allo 0.
2. Rivelatori di pressione disposti lungo la casa, con la condizione di circuito aperto corrispondente allo 0.
3. Dispositivi optoelettronici, pilotati da sorgenti all'infrarosso, per controllare i movimenti nei corridoi e nelle stanze.
4. Rivelatori di prossimità magnetici o capacitivi piazzati in posizioni strategiche lungo la casa.
5. Una combinazione dei sistemi già descritti.

L'elettronica di questo sistema è illustrata in figura 5.8.
Per controllare un sistema d'allarme esterno, inserite un optoaccoppiatore fra una qualunque delle uscite del latch e il vostro segnale d'allarme.
È possibile modificare il programma in modo da far suonare l'allarme ogni pochi secondi usando:

```
140 OUT 31, 255 : PAUSE 10 : OUT 31,0 : PAUSE 10:
    GOTO 140
```

Il programma evita i falsi allarmi causati dal volo di una mosca attraverso il raggio infrarosso o dal vento che, facendo tremare le porte, può porre intermittentemente un ingresso a valori di allarme.
La linea 100 fa sì che il programma attenda alcuni secondi, in modo da scoprire se la situazione di allarme è permanente prima di eccitare le sirene. È possibile usare il programma timer presentato nel paragrafo 4.8 assieme all'allarme, per accendere e spegnere luci in determinate stanze

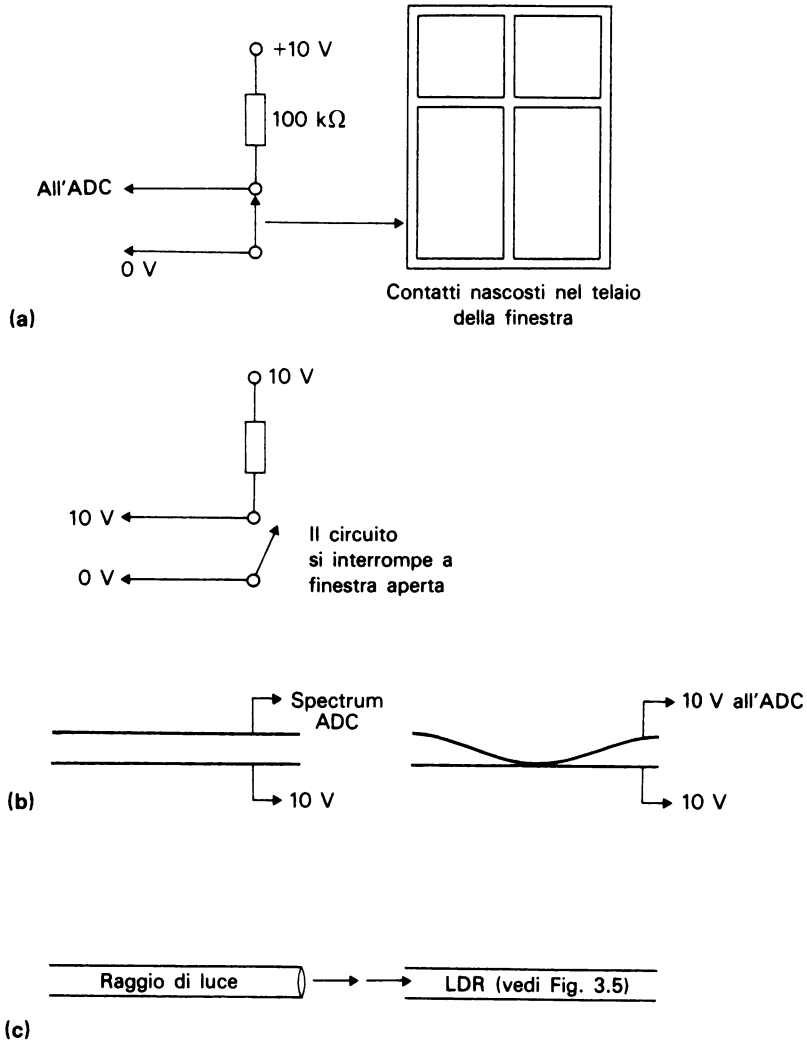


Figura 5.8 Il programma "allarme". (a) Rilevamento di porte e finestre aperte. (b) Rilevamento di intrusi per mezzo di un tappetino sensibile alla pressione. (c) Un sistema a raggi infrarossi.

al fine di scoraggiare i ladri facendo loro credere che la casa è abitata. Se, comunque, qualcuno dovesse introdursi, questo programma scoprirà senz'altro l'intruso; è ovvio che dovrete usare la massima cura nel nascondere il computer, i suoi alimentatori e le sue periferiche.

Capitolo

Altri progetti 6

Questo capitolo descrive alcuni progetti hardware che non sfruttano connessioni digitali o analogiche, ma possono semplificare la vita agli utenti dello Spectrum. I progetti vanno da un singolo pezzo di filo fino ad una tastiera completa e dotata di 40 tasti. Tutti questi progetti, comunque, per desiderabili che siano, non sono essenziali.

6.1 Come resuscitare uno Spectrum bloccato

Quando, durante l'esecuzione di un programma in codice-macchina o di un programma in BASIC contenente routine in codice-macchina, lo schermo diventa improvvisamente vuoto, la causa è da ricercarsi in errori di programmazione come loop infiniti, mancanza del comando RETURN (201), divisioni per 0, e così via.
Un filo connesso, quando serve, fra il piedino INT e lo 0 V può, contemporaneamente alla pressione dei tasti SHIFT/BREAK, riguadagnare il controllo del programma, così che l'utente possa continuare il debugging o la programmazione senza bisogno di ricaricare il programma. È buona abitudine salvare i programmi via via che essi vengono digitati per evitare la loro perdita in caso di blocco del sistema, risparmiandosi così frustrazioni e perdite di tempo.
Il cablaggio necessario è illustrato in figura 6.1. Un interruttore a pulsante normalmente aperto viene connesso tra il piedino INT e lo 0 V; quest'interruttore potrà essere alloggiato sulla basetta ADC, o sulla scheda latch, o, nel caso non si desiderasse al momento usare altra elettroni-

ca, su una piccola basetta appositamente realizzata. La procedura da seguire quando il sistema va in blocco è:

1. Premete i tasti SHIFT e BREAK come al solito
2. Premete una volta il pulsante che avete aggiunto.
3. Premete il tasto ENTER: dovreste ora ottenere il listing del programma.

È bene precisare che quest'operazione non è infallibile, ma fornisce alcune speranze di resuscitare il sistema da un blocco.

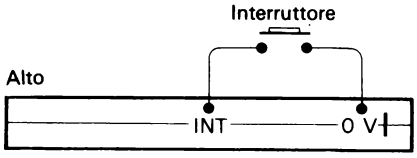


Figura 6.1 Vista posteriore dello Spectrum in cui si vedono le connessioni del pulsante antiblocco.

6.2 Un amplificatore

L'altoparlante dello Spectrum è molto piccolo e i suoni da esso generati sono praticamente inudibili in una stanza un po' rumorosa. L'uscita sonora può essere amplificata collegando i connettori EAR e MIC del registratore a cassette, commutando il registratore in riproduzione (dopo aver rimosso il nastro) e alzando il volume. Poche e semplici modifiche ad una radiolina a transistor di poco prezzo possono fornire all'utente le seguenti comodità:

1. Possibilità di amplificare il BEEP e di regolarne il volume.
2. Un segnale di monitor audio per la registrazione e la riproduzione, da usare durante i SAVE e i LOAD.
3. Un amplificatore da connettore all'uscita della basetta DAC.
4. Connettori da 3.5 mm per il collegamento permanente al registratore a cassette, eliminando la necessità di scambiare i cavetti nelle operazioni di SAVE e LOAD.
5. Una radio per rilassarvi.

Molte radioline a transistor di basso prezzo sono semplici e comprendono una basetta a circuito stampato sopra la quale sono montati i controlli di volume e di sintonia, una batteria da 9 V, un altorparlante e un po'

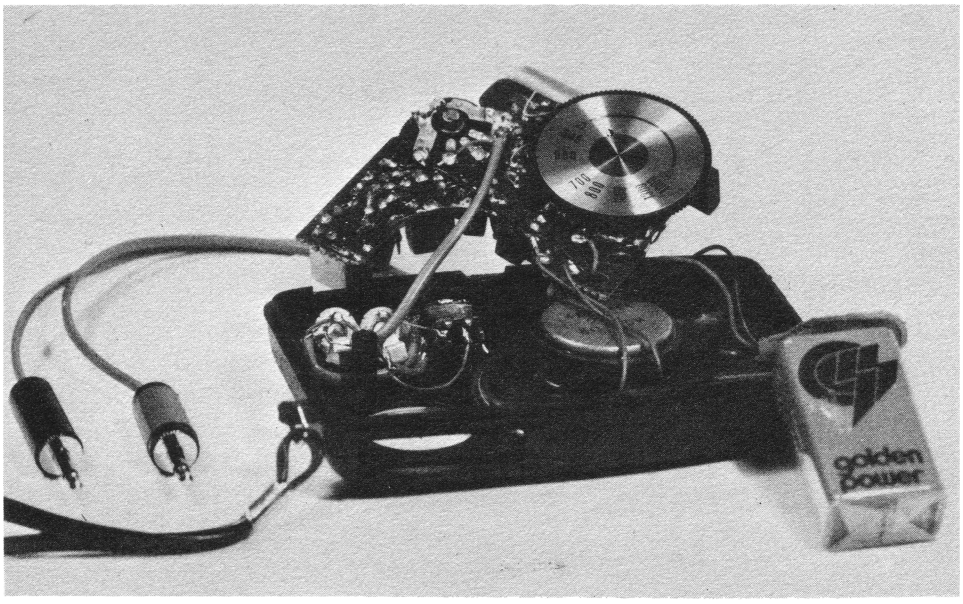


Figura 6.2 Modifiche necessarie per trasformare una radiolina a transistor in un amplificatore da usare durante i SAVE e i LOAD. Da notare le due prese jack da 3.5 mm e l'interruttorino attaccati al contenitore di plastica, i due cavetti con jack da 3.5 mm da attaccare alle prese MIC e EAR e i collegamenti al controllo del volume.

di spazio entro cui è possibile inserire le nostre modifiche. Una radio di questo tipo, modificata per funzionare da amplificatore, è presentata in figura 6.2. In figura 6.3 possiamo vedere la disposizione dei componenti all'interno di questa radiolina. Occorre montare due prese jack da 3.5 mm e un piccolo commutatore il più vicino possibile al potenziometro del volume. In figura 6.4 possiamo vedere lo schema elettrico del circuito, con i due jack da 3.5 mm per connettere gli spinotti MIC e EAR provenienti dal registratore a cassette. È inoltre necessario preparare due cavetti lunghi circa 10 cm montati con jack da 3.5 mm alle estremità per le connessioni con lo Spectrum. Il piccolo interruttore è connesso in serie al circuito EAR per evitare l'insorgere di reazioni acustiche quando tutti gli spinotti sono inseriti nelle loro sedi. Può essere etichettato SAVE/LOAD.
Infine, il segnale microfonico (proveniente dalla presa jack da 3.5 mm) viene connesso per mezzo di uno spezzone di cavo schermato al potenziometro di controllo volume della radio e saldato ai terminali estremi della pista resistiva come si vede in figura 6.3. Grazie a questo accorgimento è

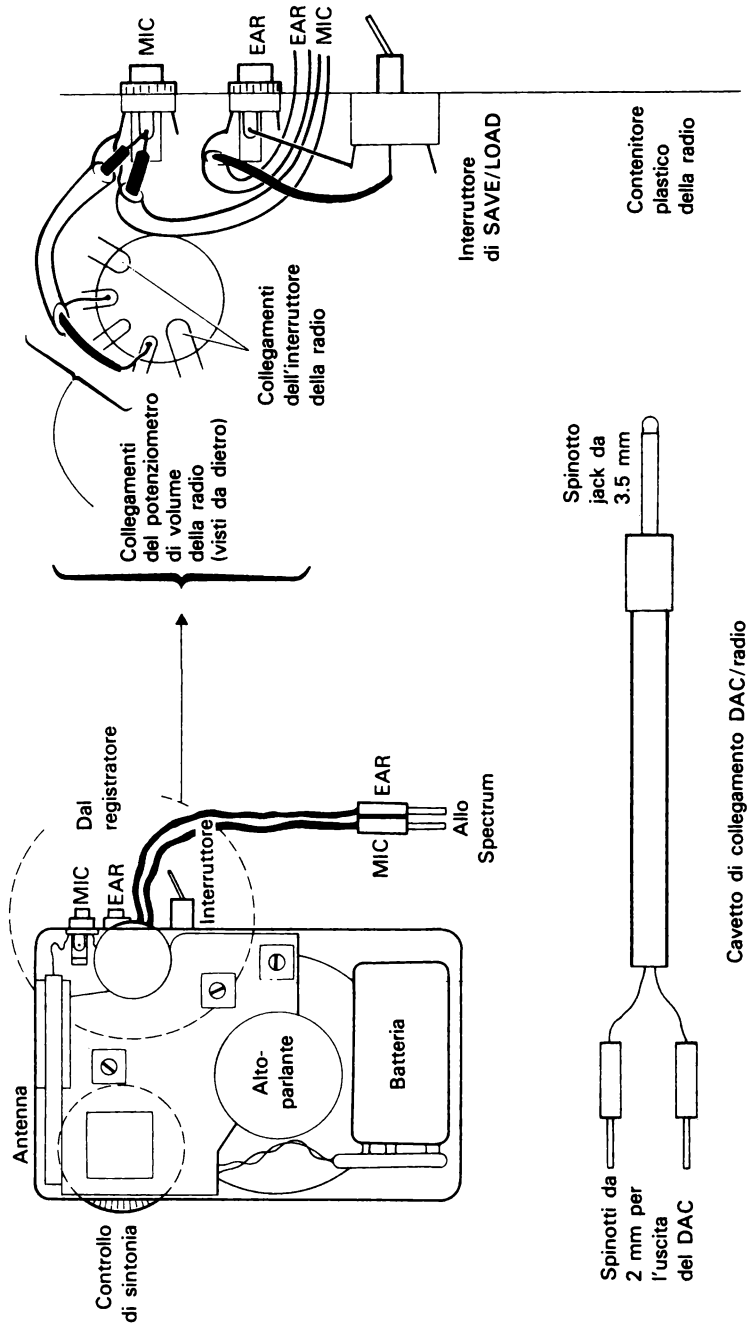


Figura 6.3. Adattamento di una radiolina tascabile come amplificatore di suono e come commutatore automatico di LOAD/SAVE.

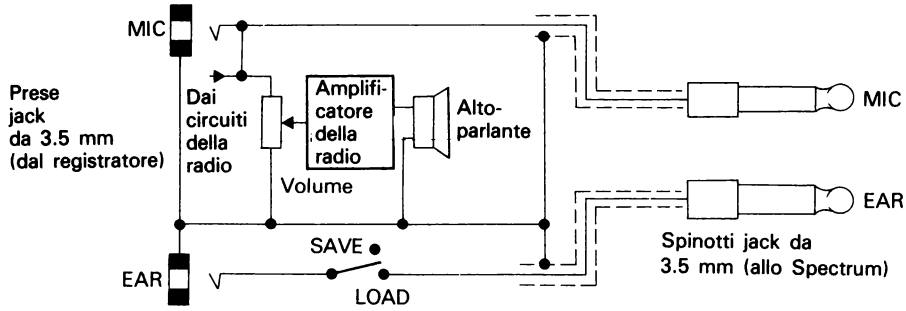


Figura 6.4 Schema elettrico dell'amplificatore.

possibile usare come di consueto il controllo di volume della radio.
I componenti vengono montati dentro al contenitore della radio nel modo che sembrerà più conveniente. Occorrerà effettuare dei fori nel contenitore attraverso cui fissare le prese jack, l'interruttore, e far passare il ca-

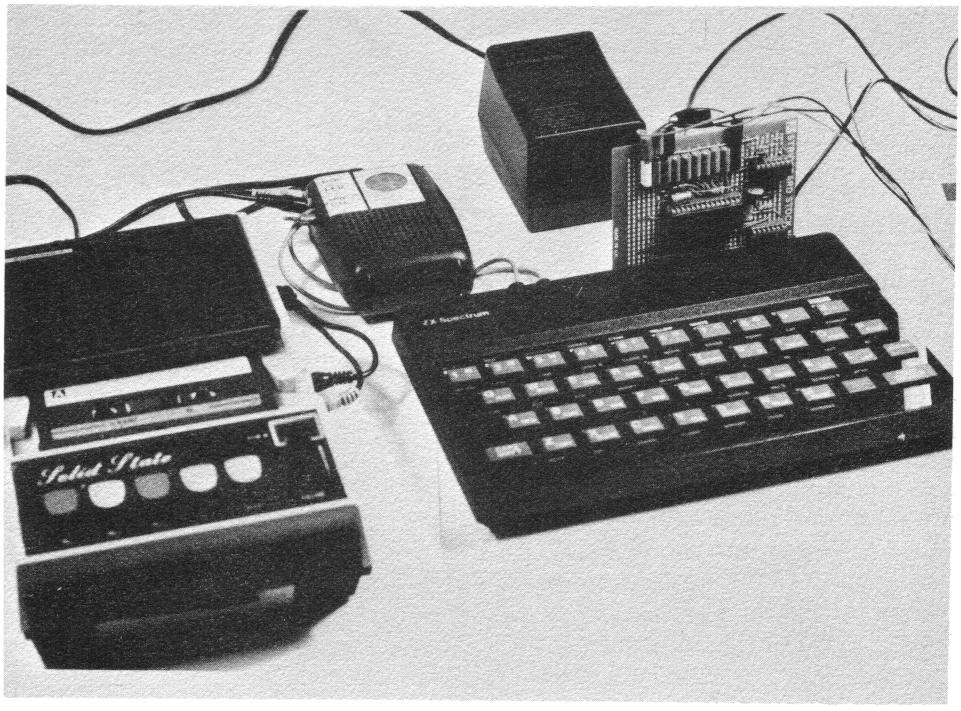


Figura 6.5 Amplificatore al lavoro con la basetta ADC.

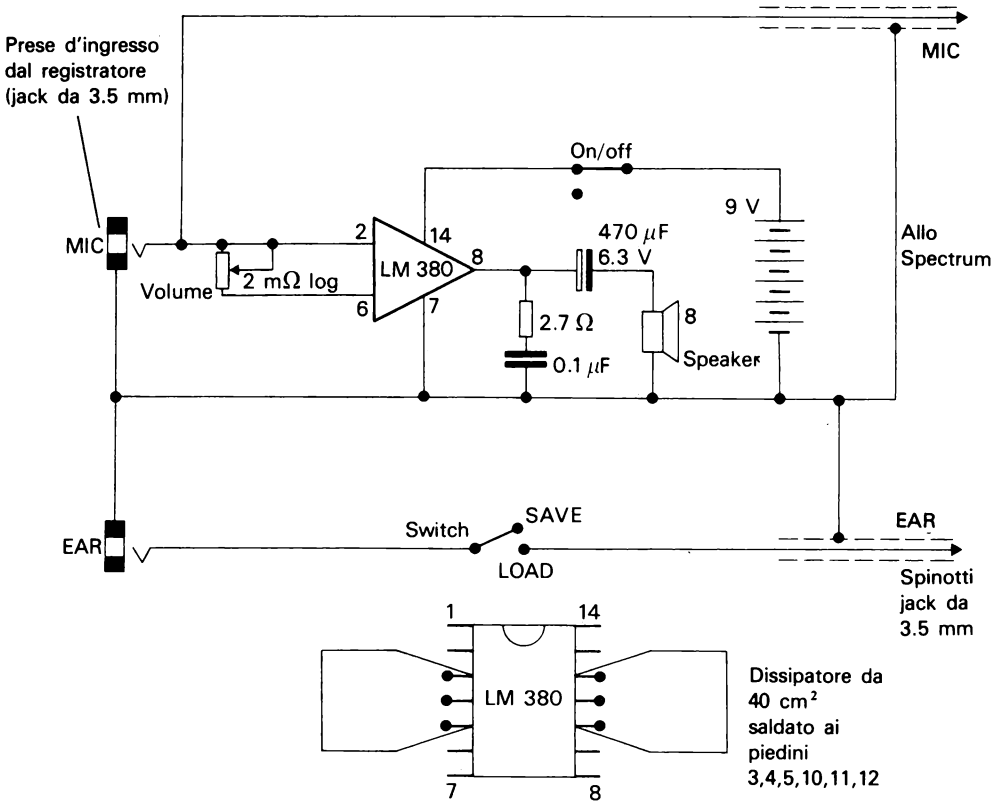


Figura 6.6 Un altro circuito amplificatore.

vetto diretto allo Spectrum. Usando la radio come amplificatore, dovreste provvedere a sintonizzarla su una parte della gamma in cui non siano presenti stazioni, salvo che non gradiate un po' di musica di sottofondo. Il sistema completo è visibile in figura 6.5.

Gli hobbisti che non desiderano cannibalizzare una radiolina portatile, possono sfruttare il circuito integrato LM380 per costruire un piccolo amplificatore adatto allo Spectrum. Il cablaggio del circuito è simile a quello già descritto per la radio.

L'integrato LM380, presentato in figura 6.6, dovrebbe essere montato su una piccola basetta a circuito stampato e dotato di un dissipatore saldato come si vede in figura. Poiché è improbabile che richiediate all'LM380 alte potenze, potreste semplicemente saldare i piedini alle piste del circuito stampato. Il controllo di volume può inglobare anche l'interruttore d'accensione e può essere montato assieme alle prese jack da 3.5 mm e

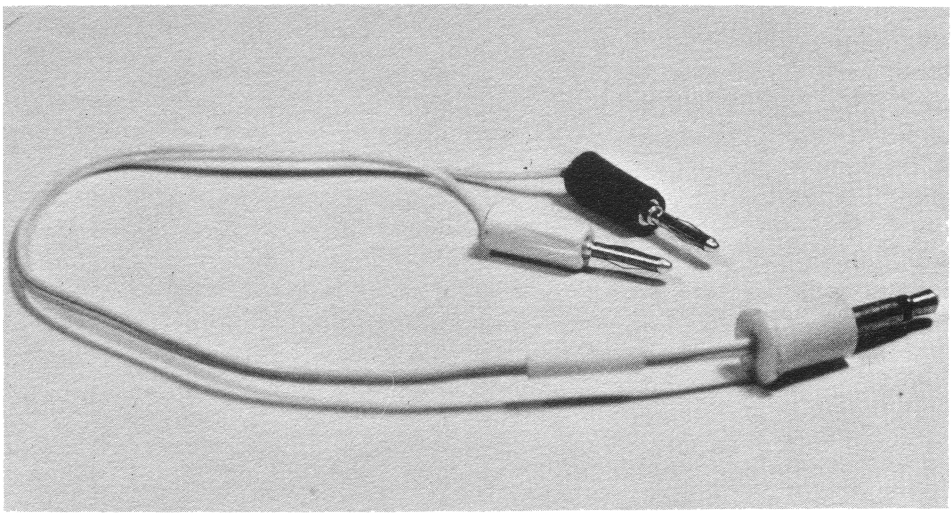


Figura 6.7 Cavetto che connette l'amplificatore con l'uscita del DAC.

all'interruttore SAVE/LOAD sul contenitore. Ricordate di lasciare spazio per una piccola batteria da 9 V, che dovrebbe durarvi per varie settimane, purché vi ricordiate di spegnere il circuito dopo l'uso. È possibile usare anche altri circuiti integrati, ma l'LM380 è stato scelto per le sue ridotte dimensioni e per il prezzo ragionevole.
Per sfruttare l'amplificatore con il circuito DAC, è necessario preparare un cavetto (come si vede in Fig. 6.7) dotato ad un estremo di un jack da 3.5 mm e all'altro estremo di due spinotti da 2 mm, da connettere all'uscita dell'amplificatore operazionale.
Un'ultima cosa: provate a tenere la radio accesa vicino allo Spectrum in azione; i suoni provenienti dalla radio forniranno un'indicazione dei milioni di operazioni che avvengono nel computer ogni secondo.

6.3 Una tastiera migliore

La tastiera dello Spectrum è superiore a quella dello ZX80 o dello ZX81, ma molti altri computer dispongono di tastiere ancora migliori. Per esempio la tastiera dello Spectrum non fornisce la sensazione tattile cui si è abituati con una macchina da scrivere: è pertanto molto facile perdere caratteri, oppure (mantenendo un tasto schiacciato molto a lungo) ripetere lo stesso carattere più volte. È estremamente facile sostituire la tastiera dello Spectrum con una migliore. Ho acquistato per pochi soldi

una tastiera supplementare e, con un minimo di lavoro, l'ho adattata al mio sistema.
La tastiera è collegata ai due connettori presenti all'interno dello Spectrum. Questo è l'unico progetto che richiede la manomissione del computer, operazione che, data la fragilità delle parti in gioco, non raccomando di eseguire spesso. I due connettori KB1 e KB2 ricevono i segnali generati dai pulsanti della tastiera connessi a matrice, come si vede in figura 6.8. Le quattro righe di tasti sono collegate, in otto gruppi di cinque, a KB2; le colonne sono invece collegate a KB1. La pressione di un tasto genera un contatto fra una riga e una colonna, che viene rilevato dalla circuiteria interna del computer. Per aggiungere l'altra tastiera dovrete:

1. Aprire il coperchio superiore dello Spectrum (la tastiera originale) e disconnetterlo da KB1 e KB2.

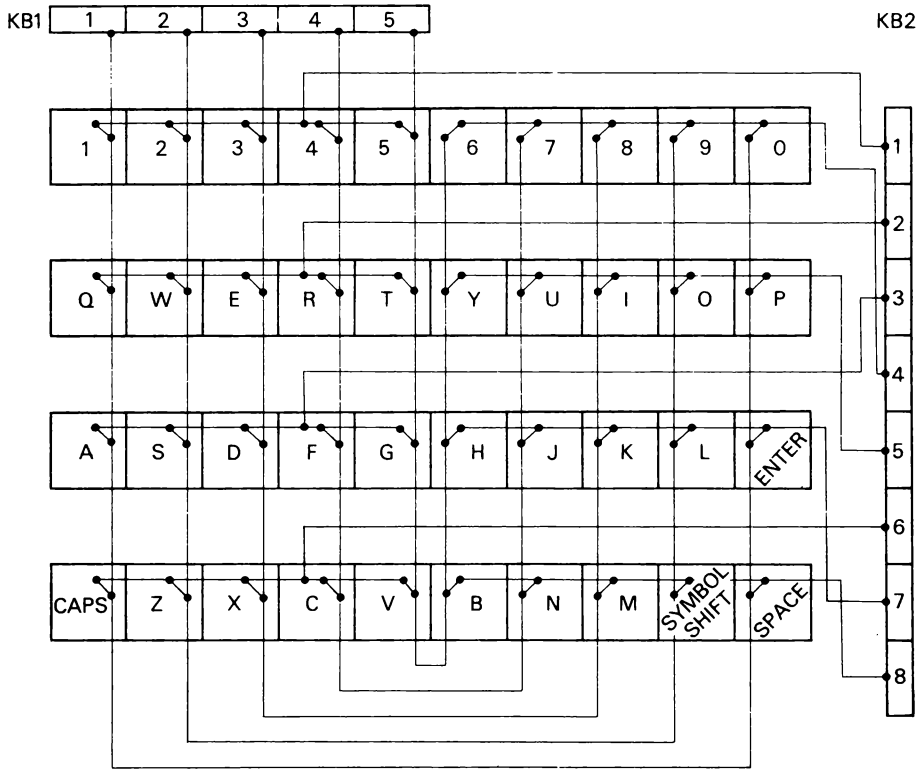


Figura 6.8 Collegamenti della tastiera.

2. Effettuare i collegamenti interni alla nuova tastiera come in figura 6.8 e connetterla a KB1 e KB2 usando cavo flessibile o multipolare con piccoli pezzi di veroboard da 1/10 di pollice saldato alla fine: questi spezzoni si inseriranno in KB1 e KB2.
3. Controllare attentamente il lavoro, correggendolo dove necessario.
4. Etichettare ogni tasto come è sullo Spectrum. È possibile usare trasferibili, o preparare una mascherina da sistemare sulla tastiera, con le funzioni assegnate stampate accanto ad ogni tasto.

La vecchia tastiera sarà ora messa da parte accuratamente imballata in caso di utilizzo futuro. È consigliabile conservare lo Spectrum dotato della nuova tastiera in un contenitore antipolvere, per proteggere la circuiteria estremamente delicata e i pulsanti della tastiera.

6.4 Lo Spectrum in automobile

Questo progetto non riguarda l'installazione di uno Spectrum nel cruscotto, anche se i convertitori analogico-digitale e digitale-analogico potrebbero avere svariati usi in un'automobile, ma è dedicato a coloro che desiderano la possibilità di usare il loro Spectrum durante i viaggi, o lontano da casa, quando l'unica sorgente di alimentazione disponibile è una batteria da 12 V, o, in una caravan, un generatore da 12 V.
Lo Spectrum richiede circa 1.2 A a 9 V. Sfortunatamente, regolatori da 9 V in grado di fornire questa corrente sono di difficile reperimento. Poiché i tipi a 5 V, 12 V, 15 V sono più comuni, si possono usare i seguenti metodi alternativi:

1. Sfruttare un regolatore a tensione d'uscita variabile (come il 317K) in grado di fornire fino a 1.5 A a tensioni di uscita comprese tra 1.2 e 37 V. Questo circuito richiede un minimo di componenti e può essere alimentato dalla batteria della macchina. Il suo schema elettrico è presentato in figura 6.9; ricordate di fornire il 317K di un idoneo dissipatore di alluminio per rimuovere il calore prodotto. Dovrebbe essere possibile usare anche regolatori variabili come il modello 78HG, ma non si può essere sicuri di ricavare da questi 9 V alimentandoli con una batteria da 12 V poiché generano cadute di tensione minime dell'ordine di 3V-4V.
2. È possibile anche usare, invece di un regolatore, un circuito che genera una caduta di tensione serie per abbassare a 9.5 V i 12 V della batteria. La più semplice realizzazione di un circuito così concepito che dia anche minori problemi di dissipazione, richiede quattro diodi al si-

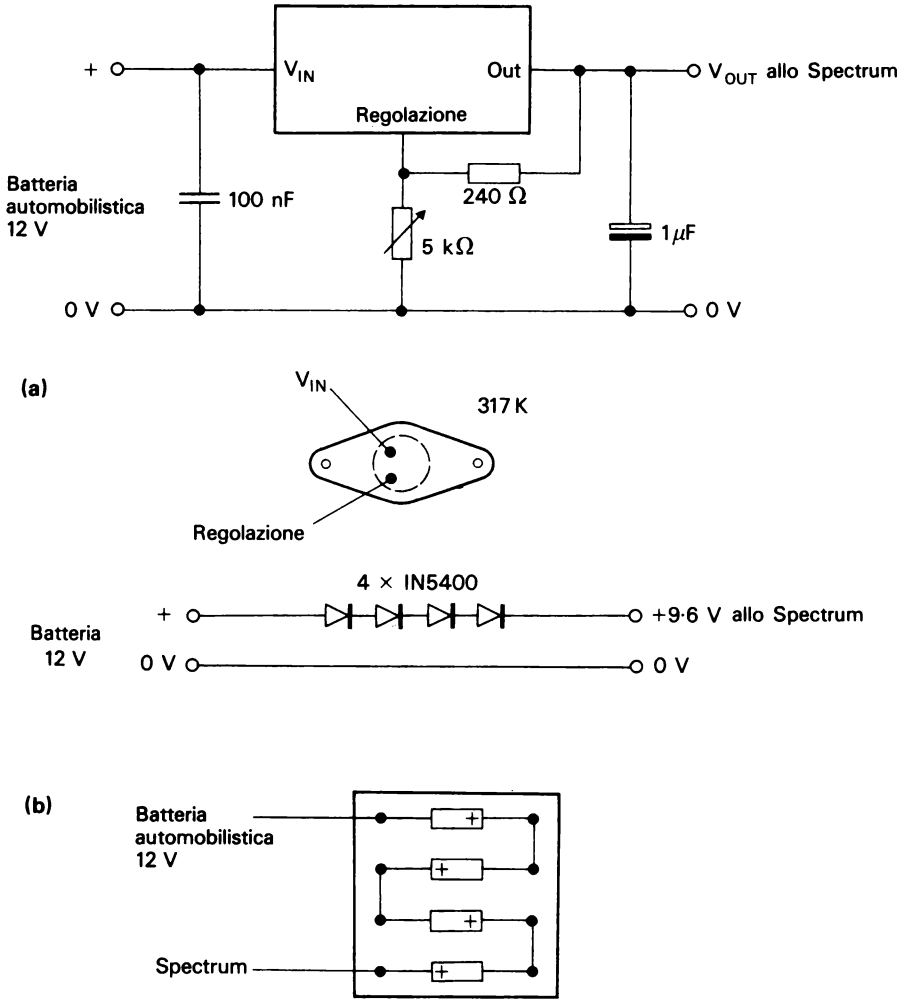


Figura 6.9 Alimentazione dello Spectrum da una batteria di automobile. (a) Con un regolatore modello 317K. (b) Usando quattro diodi per abbassare il voltaggio.

licio da 3 A (ad esempio 1N5400); questi diodi costano molto poco e formano un circuito estremamente semplice ed affidabile se connessi come in figura 6.9b. Lo Spectrum contiene i suoi propri regolatori per le varie tensioni richieste, e rende quindi non necessarie ulteriori stabilizzazioni.

È importante notare che l'uso congiunto di uno Spectrum e di un televi-

sore portatile costituirà un notevole carico per la batteria dell'automobile. Per ragioni di interferenza e di stabilità della tensione della batteria, non è consigliabile usare lo Spectrum quando il motore è acceso.

6.5 Un alimentatore d'emergenza per lo Spectrum

La rete di alimentazione domestica non è sempre totalmente affidabile. In piccoli centri e in posti lontani dalle grandi città la tensione di rete talvolta manca per brevi periodi, causando la perdita di tutti i programmi e i dati contenuti in quel momento nello Spectrum. È possibile risolvere il problema installando una batteria di tipo PP9 e un diodo 1N5400 (dello stesso tipo di quelli menzionati nel paragrafo precedente), che, connessi come in figura 6.10, forniranno i pochi minuti di alimentazione necessari per permettere all'utente di salvare il programma che ha in memoria. Non serve usare una spia di interruzione di energia, poiché ciò sarà segnalato dallo spegnimento dello schermo televisivo.

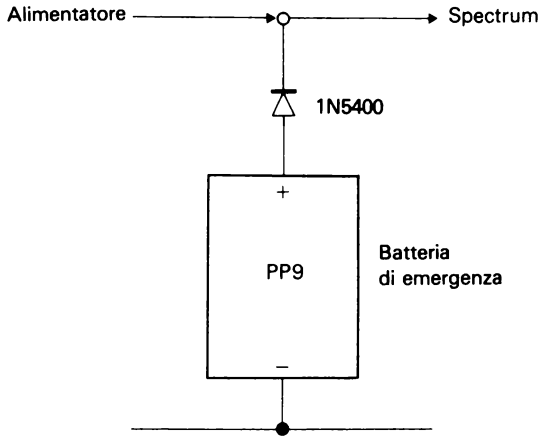


Figura 6.10 Batteria di emergenza per lo Spectrum.

In condizioni di funzionamento normale, il diodo sarà polarizzato all'inverso e quindi non condurrà; è necessario spegnere o disconnettere questo circuito quando lo Spectrum non viene usato, per evitare di fare scaricare la batteria attraverso i circuiti dello Spectrum.

Appendice

I codici-macchina dello Z80

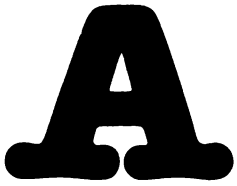


Tabella A.1 Codici-macchina (mnemonici) dello Z80

(HL) – è il numero contenuto nella locazione il cui indirizzo è puntato da HL
NN – supponendo che sia scritto come N_1N_2, N_2 è inserito nel byte basso (da 0 a 255), seguito da N_1 nel byte alto (da 0 a 255 per 256). Per esempio, il numero decimale 4000 è inserito come 3840+160=160 (N_2), 15 (N_1), poiché 15×256=3840.
Registro A – è l'accumulatore.

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
0	1	00	nop	Nessuna operazione
1	3	01	ld bc,NN	Carica nella coppia di registri BC il valore NN
2	1	02	ld (bc),a	Immagazzina l'accumulatore nella locazione di memoria indirizzata dalla coppia di registri BC
3	1	03	inc bc	Incrementa di 1 il contenuto della coppia di registri BC
4	1	04	inc b	Incrementa di 1 il contenuto del registro B
5	1	05	dec b	Diminuisce di 1 il contenuto del registro B
6	2	06	ld b,N	Carica nel registro B il valore N
7	1	07	rlca	Rotazione circolare verso sinistra dell'accumulatore
8	1	08	ex af, af	Scambia il contenuto di A e di F con quello di A' e F'
9	1	09	add hl,bc	Somma il contenuto della coppia di registri HL con quello di BC, lasciando il risultato in HL
10	3	0A	ld a,(bc)	Immagazzina nell'accumulatore il contenuto della locazione di memoria indirizzata dalla coppia di registri BC
11	1	0B	dec bc	Diminuisce di 1 il contenuto della coppia di registri BC
12	1	0C	dec c	Incrementa di 1 il contenuto del registro C
13	1	0D	dec c	Diminuisce di 1 il contenuto del registro C
14	2	0E	ld c,N	Carica nel registro C il valore N
15	1	0F	rrca	Rotazione circolare dell'accumulatore verso destra
16	2	10	djnz x	Diminuisce di 1 il contenuto del registro B, e, se il risultato è zero, salta in avanti o indietro di x
17	3	11	ld de,NN	Carica nella coppia di registri DE il valore NN
18	1	12	ld(de),a	Immagazzina l'accumulatore nella locazione di memoria indirizzata dalla coppia di registri DE

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
19	1	13	inc de	Incrementa di 1 il contenuto della coppia di registri DE
20	1	14	inc d	Incrementa di 1 il contenuto del registro D
21	1	15	dec d	Diminuisce di 1 il contenuto del registro D
22	2	16	ld d,N	Carica nel registro D il valore N
23	1	17	rla	Rotazione circolare verso sinistra dell'accumulatore attraverso il flag di carry (riporto)
24	2	18	jr x	Salta in avanti o in indietro di x
25	1	19	add hl,de	Somma il contenuto della coppia di registri HL con quello di DE, lasciando il risultato in HL
26	3	1A	ld a,(de)	Immagazzina nell'accumulatore il contenuto della locazione di memoria indirizzata dalla coppia di registri DE
27	1	1B	dec de	Diminuisce di 1 il contenuto della coppia di registri DE
28	1	1C	inc e	Incrementa di 1 il contenuto del registro E
29	1	1D	dec e	Diminuisce di 1 il contenuto del registro E
30	2	1E	ld e,N	Carica nel registro E il valore N
31	1	1F	rra	Rotazione circolare verso destra dell'accumulatore attraverso il flag di carry
32	2	20	jr nz, x	Salta in avanti o in indietro di x, se il flag Z è a zero (cioè, se è non-zero)
33	3	21	ld hl,NN	Carica nella coppia di registri HL il valore NN
34	3	22	ld (NN),hl	Immagazzina il contenuto della coppia di registri HL nella locazione NN
35	1	23	inc hl	Incrementa di 1 il contenuto della coppia di registri HL
36	1	24	inc h	Incrementa di 1 il contenuto del registro H
37	1	25	dec h	Diminuisce di 1 il contenuto del registro H
38	2	26	ld h,N	Carica nel registro H il valore N
39	1	27	daa	Aggiustamento decimale dell'accumulatore
40	2	28	jr z, x	Salta in avanti o in indietro di x, se il flag Z è a uno (cioè se è zero)
41	1	29	add hl,hl	Raddoppia il contenuto della coppia di registri HL
42	3	2A	ld hl,(NN)	Immagazzina nella coppia di registri HL il contenuto della locazione NN
43	1	2B	dec hl	Diminuisce di 1 il contenuto della coppia di registri HL
44	1	2C	inc l	Incrementa di 1 il contenuto del registro L
45	1	2D	dec l	Diminuisce di 1 il contenuto del registro L
46	2	2E	ld l,N	Carica nel registro L il valore N
47	1	2F	cpl	Complementa l'accumulatore
48	2	30	jr nc, x	Salta in avanti o in indietro di x se il flag C è a zero (cioè se è non-carry)
49	3	31	ld sp,NN	Carica nel puntatore allo stack (stack pointer) il valore NN
50	3	32	ld (NN),a	Immagazzina l'accumulatore nella locazione NN
51	1	33	inc sp	Incrementa di 1 il contenuto dello stack pointer
52	1	34	inc (hl)	Incrementa di 1 il contenuto della locazione di memoria indirizzata dalla coppia di registri HL
53	1	35	dec (hl)	Diminuisce di 1 il contenuto della locazione di memoria indirizzata dalla coppia di registri HL
54	2	36	ld (hl),N	Carica il valore N nella locazione di memoria indirizzata dalla coppia di registri HL
55	1	37	scf	Pone a uno il flag C
56	2	38	jr c, x	Salta in avanti o in indietro di x se il flag C è a uno (cioè se è carry)
57	1	39	add hl,sp	Somma al contenuto della coppia di registri HL il contenuto del registro SP (stack pointer) lasciando il risultato in HL
58	3	3A	ld a,(NN)	Immagazzina nell'accumulatore il contenuto della locazione NN
59	1	3B	dec sp	Diminuisce di 1 il contenuto del registro SP (stack pointer)
60	1	3C	inc a	Incrementa di 1 il contenuto dell'accumulatore

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
61	1	3D	dec a	Diminuisce di 1 il contenuto dell'accumulatore
62	2	3E	ld a,N	Immagazzina nell'accumulatore il valore N
63	1	3F	ccf	Completa il flag C
64	1	40	ld b,b	Copia il contenuto del registro B nel registro B
65	1	41	ld b,c	Copia il contenuto del registro C nel registro B
66	1	42	ld b,d	Copia il contenuto del registro D nel registro B
67	1	43	ld b,e	Copia il contenuto del registro E nel registro B
68	1	44	ld b,h	Copia il contenuto del registro H nel registro B
69	1	45	ld b,l	Copia il contenuto del registro L nel registro B
70	1	46	ld b,(hl)	Carica il registro B dalla locazione di memoria indirizzata dalla coppia di registri HL
71	1	47	ld b,a	Copia il contenuto dell'accumulatore nel registro B
72	1	48	ld c,b	Copia il contenuto del registro B nel registro C
73	1	49	ld c,c	Copia il contenuto del registro C nel registro C
74	1	4A	ld c,d	Copia il contenuto del registro D nel registro C
75	1	4B	ld c,e	Copia il contenuto del registro E nel registro C
76	1	4C	ld c,h	Copia il contenuto del registro H nel registro C
77	1	4D	ld c,l	Copia il contenuto del registro L nel registro C
78	1	4E	ld c,(hl)	Carica il registro C dalla locazione di memoria indirizzata dalla coppia di registri HL
79	1	4F	ld c,a	Copia il contenuto dell'accumulatore nel registro C
80	1	50	ld d,b	Copia il contenuto del registro B nel registro D
81	1	51	ld d,c	Copia il contenuto del registro C nel registro D
82	1	52	ld d,d	Copia il contenuto del registro D nel registro D
83	1	53	ld d,e	Copia il contenuto del registro E nel registro D
84	1	54	ld d,h	Copia il contenuto del registro H nel registro D
85	1	55	ld d,l	Copia il contenuto del registro L nel registro D
86	1	56	ld d,(hl)	Carica il registro D dalla locazione di memoria indirizzata dalla coppia di registri HL
87	1	57	ld d,a	Copia il contenuto dell'accumulatore nel registro D
88	1	58	ld e,b	Copia il contenuto del registro B nel registro E
89	1	59	ld e,c	Copia il contenuto del registro C nel registro E
90	1	5A	ld e,d	Copia il contenuto del registro D nel registro E
91	1	5B	ld e,e	Copia il contenuto del registro E nel registro E
92	1	5C	ld e,h	Copia il contenuto del registro H nel registro E
93	1	5D	ld e,l	Copia il contenuto del registro L nel registro E
94	1	5E	ld e,(hl)	Carica il registro E dalla locazione di memoria indirizzata dalla coppia di registri HL
95	1	5F	ld e,a	Copia il contenuto dell'accumulatore nel registro E
96	1	60	ld h,b	Copia il contenuto del registro B nel registro H
97	1	61	ld h,c	Copia il contenuto del registro C nel registro H
98	1	62	ld h,d	Copia il contenuto del registro D nel registro H
99	1	63	ld h,e	Copia il contenuto del registro E nel registro H
100	1	64	ld h,h	Copia il contenuto del registro H nel registro H
101	1	65	ld h,l	Copia il contenuto del registro L nel registro H
102	1	66	ld h,(hl)	Carica il registro H dalla locazione di memoria indirizzata dalla coppia di registri HL
103	1	67	ld h,a	Copia il contenuto dell'accumulatore nel registro H
104	1	68	ld l,b	Copia il contenuto del registro B nel registro L
105	1	69	ld l,c	Copia il contenuto del registro C nel registro L
106	1	6A	ld l,d	Copia il contenuto del registro D nel registro L
107	1	6B	ld l,e	Copia il contenuto del registro E nel registro L
108	1	6C	ld l,h	Copia il contenuto del registro H nel registro L
109	1	6D	ld l,l	Copia il contenuto del registro L nel registro L
110	1	6E	ld l,(hl)	Carica il registro L dalla locazione di memoria indirizzata dalla coppia di registri HL
111	1	6F	ld l,a	Copia il contenuto dell'accumulatore nel registro L

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
112	1	70	ld (hl),b	Copia il contenuto del registro B nella locazione indirizzata da HL
113	1	71	ld (hl),c	Copia il contenuto del registro C nella locazione indirizzata da HL
114	1	72	ld (hl),d	Copia il contenuto del registro D nella locazione indirizzata da HL
115	1	73	ld (hl),e	Copia il contenuto del registro E nella locazione indirizzata da HL
116	1	74	ld (hl),h	Copia il contenuto del registro H nella locazione indirizzata da HL
117	1	75	ld (hl),l	Copia il contenuto del registro H nella locazione indirizzata da HL
118	1	76	halt	Blocca la CPU
119	1	77	ld (hl),a	Copia il contenuto dell'accumulatore nella locazione indirizzata da HL
120	1	78	ld a,b	Copia il contenuto del registro B nell'accumulatore
121	1	79	ld a,c	Copia il contenuto del registro C nell'accumulatore
122	1	7A	ld a,d	Copia il contenuto del registro D nell'accumulatore
123	1	7B	ld a,e	Copia il contenuto del registro E nell'accumulatore
124	1	7C	ld a,h	Copia il contenuto del registro H nell'accumulatore
125	1	7D	ld a,l	Copia il contenuto del registro L nell'accumulatore
126	1	7E	ld a,(hl)	Carica l'accumulatore dalla locazione di memoria indirizzata dalla coppia di registri HL
127	1	7F	ld a,a	Copia il contenuto dell'accumulatore nell'accumulatore
128	1	80	add a,b	B+A→A
129	1	81	add a,c	C+A→A
130	1	82	add a,d	D+A→A
131	1	83	add a,e	E+A→A
132	1	84	add a,h	H+A→A
133	1	85	add a,l	L+A→A
134	1	86	add a,(hl)	(HL)+A→A
135	1	87	add a,a	A+A→A
136	1	88	adc a,b	B+A+carry→A
137	1	89	adc a,c	C+A+carry→A
138	1	8A	adc a,d	D+A+carry→A
139	1	8B	adc a,e	E+A+carry→A
140	1	8C	adc a,h	H+A+carry→A
141	1	8D	adc a,l	L+A+carry→A
142	1	8E	adc a,(hl)	(HL)+A+carry→A
143	1	8F	adc a,a	A+A+carry→A
144	1	90	sub b	A−B→A
145	1	91	sub c	A−C→A
146	1	92	sub d	A−D→A
147	1	93	sub e	A−E→A
148	1	94	sub h	A−H→A
149	1	95	sub l	A−L→A
150	1	96	sub (hl)	A−(HL)→A
151	1	97	sub a	A−A→A
152	1	98	sbc a,b	A−B−carry→A
153	1	99	sbc a,c	A−C−carry→A
154	1	9A	sbc a,d	A−D−carry→A
155	1	9B	sbc a,e	A−E−carry→A
156	1	9C	sbc a,h	A−H−carry→A
157	1	9D	sbc a,l	A−L−carry→A
158	1	9E	sbc a,(hl)	A−(HL)−carry→A
159	1	9F	sbc a,a	A−A−carry→A
160	1	A0	and b	A and B→A

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
161	1	A1	and c	A and C→A
162	1	A2	and d	A and D→A
163	1	A3	and e	A and E→A
164	1	A4	and h	A and H→A
165	1	A5	and l	A and L→A
166	1	A6	and (hl)	A and (HL)→A
167	1	A7	and a	A and A→A
168	1	A8	xor b	A exclusive or B→A
169	1	A9	xor c	A exclusive or C→A
170	1	AA	xor d	A exclusive or D→A
171	1	AB	xor e	A exclusive or E→A
172	1	AC	xor h	A exclusive or H→A
173	1	AD	xor l	A exclusive or L→A
174	1	AE	xor (hl)	A exclusive or (HL)−A
175	1	AF	xor a	A exclusive or A→A
176	1	B0	or b	A or B→A
177	1	B1	or c	A or C→A
178	1	B2	or d	A or D→A
179	1	B3	or e	A or E→A
180	1	B4	or h	A or H→A
181	1	B5	or l	A or L→A
182	1	B6	or(hl)	A or (HL)→A
183	1	B7	or a	A or A→A
184	1	B8	cp b	Sottrae il contenuto del registro B dall'accumulatore, scartando il risultato
185	1	B9	cp c	Sottrae il contenuto del registro C dall'accumulatore, scartando il risultato
186	1	BA	cp d	Sottrae il contenuto del registro D dall'accumulatore, scartando il risultato
187	1	BB	cp e	Sottrae il contenuto del registro E dall'accumulatore, scartando il risultato
188	1	BC	cp h	Sottrae il contenuto del registro H dall'accumulatore, scartando il risultato
189	1	BD	cp l	Sottrae il contenuto del registro L dall'accumulatore, scartando il risultato
190	1	BE	cp (hl)	Sottrae dall'accumulatore il contenuto della locazione indirizzata dalla coppia di registri HL, scartando il risultato (cioè lasciando invariato il contenuto dell'accumulatore)
191	1	BF	cp a	Sottrae il contenuto dell'accumulatore, scartando il risultato
192	1	C0	ret nz	Ritorna se il flag Z è a zero (cioè se è non-zero)
193	1	C1	pop bc	Carica dallo stack la coppia di registri BC
194	3	C2	jp nz,NN	Salta alla locazione NN se il flag Z è a zero
195	3	C3	jp NN	Salta alla locazione NN
196	3	C4	call nz,NN	Chiama la routine alla locazione NN se il flag Z è a zero
197	1	C5	push bc	Salva nello stack il contenuto della coppia di registri BC
198	2	C6	add a,N	A+N→A
199	1	C7	rst 0	Chiama la routine iniziante alla locazione 0000
200	1	C8	ret z	Ritorna se il flag Z è a uno (cioè se è zero)
201	1	C9	ret	Ritorna
202	3	CA	jp z,NN	Salta alla locazione NN se il flag Z è a uno
203		CB		Vedi l'insieme di routine CB
204	3	CC	call z,NN	Chiama la routine alla locazione NN se il flag Z è a uno
205	3	CD	call NN	Chiama la routine alla locazione NN
207	1	CF	rst 8	Chiama la routine di errore all'indirizzo 0008
208	1	D0	ret nc	Ritorna se il flag C è a zero (cioè se non-carry)
209	1	D1	pop de	Carica dallo stack la coppia di registri DE
210	3	D2	jp nc,NN	Salta alla locazione NN se il flag C è a zero

Decimale	*Byte*	*Es.*	*Codifica*	*Descrizione*
211	2	D3	out(N),a	Emette il contenuto dell'accumulatore al port N
212	3	D4	call nc,NN	Chiama la routine alla locazione NN se il flag C è a zero
213	1	D5	push de	Salva nello stack il contenuto della coppia di registri DE
214	2	D6	sub N	A−N→A
215	1	D7	rst 16	Chiama la routine di stampa all'indirizzo 0010
216	1	D8	ret c	Ritorna se il flag C è a uno
217	1	D9	exx	Scambia i registri principali con quelli alternativi
218	3	DA	jp c,NN	Salta alla locazione NN se il flag C è a uno
219	2	DB	in a,(N)	Carica l'accumulatore dal port N
220	3	DC	call c,NN	Chiama la routine alla locazione NN se il flag C è a uno
221		DD	[prefissi delle istruzioni che usano ix]	Vedi Tabella A.2
222	2	DE	sbc a,N	A−N−carry→A
223	1	DF	rst 24	Chiama la routine di carattere all'indirizzo 0018
224	1	E0	ret po	Ritorna se il flag P/O è a zero
225	1	E1	pop hl	Carica dallo stack la coppia di registri HL
226	3	E2	jp po,NN	Salta alla locazione NN se il flag P/O è a zero
227	1	E3	ex(sp),hl	Scambia il contenuto del registro SP con quello della coppia di registri HL
228	3	E4	call po,NN	Chiama la routine all'indirizzo NN se il flag P/O è a zero
229	1	E5	push hl	Salva nello stack il contenuto della coppia di registri HL
230	2	E6	and N	A and N→A
231	1	E7	rst 32	Chiama la routine di carattere alla locazione 0020
232	1	E8	ret pe	Ritorna se il flag P/O è a uno
233	1	E9	jp (hl)	Salta alla locazione indirizzata dalla coppia di registri HL
234	3	EA	jp pe,NN	Salta alla locazione NN se il flag P/O è a uno
235	1	EB	ex de,hl	Scambia il contenuto della coppia di registri DE con quello della coppia HL
236	3	EC	call pe,NN	Chiama la routine all'indirizzo NN se il flag P/O è a uno
237		ED		Vedi l'insieme di routine ED
238	2	EE	xor N	A exclusive or N→A
239	1	EF	rst 40	Chiama la routine calcolatore all'indirizzo 0028
240	1	F0	ret p	Ritorna se il flag P è a zero
241	1	F1	pop af	Carica dallo stack la coppia di registri AF
242	3	F2	jp p,NN	Salta alla locazione NN se il flag P è a zero
243	1	F3	di	Disabilita gli interrupt
244	3	F4	call p,NN	Chiama la routine all'indirizzo NN se il flag P è a zero
245	1	F5	push af	Salva nello stack il contenuto della coppia di registri AF
246	2	F6	or N	A or N→A
247	1	F7	rst 48	Chiama la routine spazio di lavoro all'indirizzo 0030
248	1	F8	ret m	Ritorna se il flag P è a uno
249	1	F9	ld sp,hl	Copia il contenuto del registro HL nel registro SP (stack pointer)
250	3	FA	jp m,NN	Salta alla locazione NN se il flag P è a uno
251	1	FB	ei	Abilita gli interrupt
252	3	FC	call m,NN	Chiama la routine all'indirizzo NN se il flag P è a uno
253		FD	[prefissi delle istruzioni che usano iy]	Vedi Tabella A.2
254	2	FE	cp N	Sottrae al contenuto dell'accumulatore il valore N scaricando il risultato
255	1	FF	rst 56	Chiama la routine all'indirizzo 0038

Tabella A.2 Codici utilizzati con DD e FD

d – significa spostamento (displacement): dd e dddd significano rispettivamente numeri di uno e due byte.
La tabella si riferisce ai mnemonici DD (221 decimale) in relazione col registro IX.
I mnemonici FD (253 decimale) usano operazioni simili, ma in relazione col registro IY.

DD 09	ADD	IX,BC	DD CB d 06	RLC	(IX+d)
DD 19	ADD	IX,DE	DD CB d 0E	RRC	(IX+d)
DD 21 +dddd	LD	IX,+dddd	DD CB d 16	RL	(IX+d)
DD 22 addr	LD	(addr),IX	DD CB d 1E	RR	(IX+d)
DD 23	INC	IX	DD CB d 26	SLA	(IX+d)
DD 29	ADD	IX,IX	DD CB d 2E	SRA	(IX+d)
DD 2A addr	LD	IX,(addr)	DD CB d 3E	SRL	(IX+d)
DD 2B	DEC	IX	DD CB d 46	BIT	0,(IX+d)
DD 34 d	INC	(IX+d)	DD CB d 4E	BIT	1,(IX+d)
DD 35 d	DEC	(IX+d)	DD CB d 56	BIT	2,(IX+d)
DD 36 d +dd	LD	(IX+d),+dd	DD CB d 5E	BIT	3,(IX+d)
DD 39	ADD	IX,SP	DD CB d 66	BIT	4,(IX+d)
DD 46 d	LD	B,(IX+d)	DD CB d 6E	BIT	5,(IX+d)
DD 4E d	LD	C,(IX+d)	DD CB d 76	BIT	6,(IX+d)
DD 56 d	LD	D,(IX+d)	DD CB d 7E	BIT	7,(IX+d)
DD 5E d	LD	E,(IX+d)	DD CB d 86	RES	0,(IX+d)
DD 66 d	LD	H,(IX+d)	DD CB d 8E	RES	1,(IX+d)
DD 6E d	LD	L,(IX+d)	DD CB d 96	RES	2,(IX+d)
DD 70 d	LD	(IX+d),B	DD CB d 9E	RS	3,(IX+d)
DD 71 d	LD	(IX+d),C	DD CB d A6	RES	4,(IX+d)
DD 72 d	LD	(IX+d),D	DD CB d AE	RES	5,(IX+d)
DD 73 d	LD	(IX+d),E	DD CB d B6	RES	6,(IX+d)
DD 74 d	LD	(IX+d),H	DD CB d BE	RES	7,(IX+d)
DD 75 d	LD	(IX+d),L	DD CB d C6	SET	0,(IX+d)
DD 77 d	LD	(IX+d),A	DD CB d CE	SET	1,(IX+d)
DD 7E d	LD	A,(IX+d)	DD CB d D6	SET	2,(IX+d)
	ADD	A,(IX+d)	DD CB d DE	SET	3,(IX+d)
DD 8E d	ADC	A,(IX+d)	DD CB d E6	SET	4,(IX+d)
DD 96 d	SUB	(IX+d)	DD CB d EE	SET	5,(IX+d)
DD 9E d	SBC	A,(IX+d)	DD CB d F6	SET	6,(IX+d)
DD A6 d	AND	(IX+d)	DD CB d FE	SET	7,(IX+d)
DD AE d	XOR	(IX+d)	DD E1	POP	IX
DD B6 d	OR	(IX+d)	DD E3	EX	(SP),IX
DD BE d	CP	(IX+d)	DD E5	PUSH	IX
			DD E9	JP	(IX)
			DD F9	LD	SP,IX

Tabella A.3 Istruzioni ED (237 decimale)

ED 40 IN B,(C)	ED 50 IN D,(C)	ED 60 IN H,(C)		ED A0 LDI	ED B0 LDIR
ED 41 OUT (C),B	ED 51 OUT (C),D	ED 61 OUT (C),H		ED A1 CPI	ED B1 CPIR
ED 42 SBC HL,BC	ED 52 SBC HL,DE	ED 62 SBC HL,HL	ED 72 SBC HL,SP	ED A2 INI	ED B2 INIR
ED 43 (addr),BC	ED 53 (addr),DE	ED 63 (addr),HL	ED 73 (addr),SP	ED A3 OUTI	ED B3 OTIR
ED 44 NEG					
ED 45 RETN					
ED 46 IM 0	ED 56 IM 1	ED 66 IM 2			
ED 47 LD I,A	ED 57 LD A,I	ED 67 RRD			
ED 48 IN C,(C)	ED 58 IN E,(C)	ED 68 IN L,(C)	ED 78 IN A,(C)	ED A8 LDD	ED B8 LDDR
ED 49 OUT (C),C	ED 59 OUT (C),E	ED 69 OUT (C),L	ED 79 OUT (C),A	ED A9 CPD	ED B9 CPDR
ED 4A ADC HL,BC	ED 5A ADC HL,DE	ED 6A ADC HL,HL	ED 7A ADC HL,SP	ED AA IND	ED BA INDR
ED 4B LD BC,(addr)	ED 5B LD DE,(addr)	ED 6B LD HL,(addr)	ED 7B LD SP,(addr)	ED AB OUTD	ED BB OTRD
ED 4D RETI					
ED 4F LD R,A	ED 5F LD A,R	ED 6F RLD			

Appendice

Tabelle di conversione esadecimali-decimali

B

Le due tabelle riportano la conversione tra valori decimali ed esadecimali, rispettivamente per i byte bassi (valori decimali da 0 a 255) e per i byte alti (valori decimali da 0 a 65280), corrispondenti a valori esadecimali da 00 a FF.
Nella prima, per valori decimali superiori a 127, viene fornito anche il complemento a due.

Tabella B.1 Decimali da 0 a 255 — Esadecimali da 00 a FF, byte basso – Complemento a 2

Dec.	Esad.	Dec.	Esad.	Dec.	Esad.	C.a2	Dec.	Esad.	C.a 2
0	00	64	40	128	80	−128	192	C0	−64
1	01	65	41	129	81	−127	193	C1	−63
2	02	66	42	130	82	−126	194	C2	−62
3	03	67	43	131	83	−125	195	C3	−61
4	04	68	44	132	84	−124	196	C4	−60
5	05	69	45	133	85	−123	197	C5	−59
6	06	70	46	134	86	−122	198	C6	−58
7	07	71	47	135	87	−121	199	C7	−57
8	08	72	48	136	88	−120	200	C8	−56
9	09	73	49	137	89	−119	201	C9	−55
10	0A	74	4A	138	8A	−118	202	CA	−54
11	0B	75	4B	139	8B	−117	203	CB	−53
12	0C	76	4C	140	8C	−116	204	CC	−52
13	0D	77	4D	141	8D	−115	205	CD	−51
14	0E	78	4E	142	8E	−114	206	CE	−50
15	0F	79	4F	143	8F	−113	207	CF	−49
16	10	80	50	144	90	−112	208	D0	−48
17	11	81	51	145	91	−111	209	D1	−47
18	12	82	52	146	92	−110	210	D2	−46
19	13	83	53	147	93	−109	211	D3	−45
20	14	84	54	148	94	−108	212	D4	−44
21	15	85	55	149	95	−107	213	D5	−43
22	16	86	56	150	96	−106	214	D6	−42
23	17	87	57	151	97	−105	215	D7	−41
24	18	88	58	152	98	−104	216	D8	−40
25	19	89	59	153	99	−103	217	D9	−39
26	1A	90	5A	154	9A	−102	218	DA	−38
27	1B	91	5B	155	9B	−101	219	DB	−37
28	1C	92	5C	156	9C	−100	220	DC	−36
29	1D	93	5D	157	9D	−99	221	DD	−35
30	1E	94	5E	158	9E	−98	222	DE	−34
31	1F	95	5F	159	9F	−97	223	DF	−33
32	20	96	60	160	A0	−96	224	E0	−32
33	21	97	61	161	A1	−95	225	E1	−31
34	22	98	62	162	A2	−94	226	E2	−30
35	23	99	63	163	A3	−93	227	E3	−29
36	24	100	64	164	A4	−92	228	E4	−28
37	25	101	65	165	A5	−91	229	E5	−27
38	26	102	66	166	A6	−90	230	E6	−26
39	27	103	67	167	A7	−89	231	E7	−25
40	28	104	68	168	A8	−88	232	E8	−24
41	29	105	69	169	A9	−87	233	E9	−23
42	2A	106	6A	170	AA	−86	234	EA	−22
43	2B	107	6B	171	AB	−85	235	EB	−21
44	2C	108	6C	172	AC	−84	236	EC	−20
45	2D	109	6D	173	AD	−83	237	ED	−19
46	2E	110	6E	174	AE	−82	238	EE	−18
47	2F	111	6F	175	AF	−81	239	EF	−17
48	30	112	70	176	B0	−80	240	F0	−16
49	31	113	71	177	B1	−79	241	F1	−15
50	32	114	72	178	B2	−78	242	F2	−14
51	33	115	73	179	B3	−77	243	F3	−13
52	34	116	74	180	B4	−76	244	F4	−12
53	35	117	75	181	B5	−75	245	F5	−11
54	36	118	76	182	B6	−74	246	F6	−10
55	37	119	77	183	B7	−73	247	F7	−9
56	38	120	78	184	B8	−72	248	F8	−8
57	39	121	79	185	B9	−71	249	F9	−7
58	3A	122	7A	186	BA	−70	250	FA	−6
59	3B	123	7B	187	BB	−69	251	FB	−5
60	3C	124	7C	188	BC	−68	252	FC	−4
61	3D	125	7D	189	BD	−67	253	FD	−3
62	3E	126	7E	190	BE	−66	254	FE	−2
63	3F	127	7F	191	BF	−65	255	FF	−1

Tabella B.2 Decimali da 0 a 54280 — Esadecimali da 00 a FF, byte alto

Decimali	Esad.	Decimali	Esad.	Decimali	Esad.	Decimali	Esad.
0	00	16 384	40	32 768	80	49 152	C0
256	01	16 640	41	33 024	81	49 408	C1
512	02	16 896	42	33 280	82	49 664	C2
768	03	17 152	43	33 536	83	49 920	C3
1 024	04	17 408	44	33 792	84	50 176	C4
1 280	05	17 664	45	34 048	85	50 432	C5
1 536	06	17 920	46	34 304	86	50 688	C6
1 792	07	18 176	47	34 560	87	50 944	C7
2 048	08	18 432	48	34 816	88	51 200	C8
2 304	09	18 688	49	35 072	89	51 456	C9
2 560	0A	18 944	4A	35 328	8A	51 712	CA
2 816	0B	19 200	4B	35 584	8B	51 968	CB
3 072	0C	19 456	4C	35 840	8C	52 224	CC
3 328	0D	19 712	4D	36 096	8D	52 480	CD
3 584	0E	19 968	4E	36 352	8E	52 736	CE
3 840	0F	20 224	4F	36 608	8F	52 992	CF
4 096	10	20 480	50	36 864	90	53 248	D0
4 352	11	20 736	51	37 120	91	53 504	D1
4 608	12	20 992	52	37 376	92	53 760	D2
4 864	13	21 248	53	37 632	93	54 016	D3
5 120	14	21 504	54	37 888	94	54 272	D4
5 376	15	21 760	55	38 144	95	54 528	D5
5 632	16	22 016	56	38 400	96	54 784	D6
5 888	17	22 272	57	38 656	97	55 040	D7
6 144	18	22 528	58	38 912	98	55 296	D8
6 400	19	22 784	59	39 168	99	55 552	D9
6 656	1A	23 040	5A	39 424	9A	55 808	DA
6 912	1B	23 296	5B	39 680	9B	56 064	DB
7 168	1C	23 552	5C	39 936	9C	56 320	DC
7 424	1D	23 808	5D	40 192	9D	56 576	DD
7 680	1E	24 064	5E	40 448	9E	56 832	DE
7 936	1F	24 320	5F	40 704	9F	57 088	DF
8 192	20	24 576	60	40 960	A0	57 344	E0
8 448	21	24 832	61	41 216	A1	57 600	E1
8 704	22	25 088	62	41 472	A2	57 856	E2
8 960	23	25 344	63	41 728	A3	58 112	E3
9 216	24	25 600	64	41 984	A4	58 368	E4
9 472	25	25 856	65	42 240	A5	58 624	E5
9 728	26	26 112	66	42 496	A6	58 880	E6
9 984	27	26 368	67	42 752	A7	59 136	E7
10 240	28	26 624	68	43 008	A8	59 392	E8
10 496	29	26 880	69	43 264	A9	59 648	E9
10 752	2A	27 136	6A	43 520	AA	59 904	EA
11 008	2B	27 392	6B	43 776	AB	60 160	EB
11 264	2C	27 648	6C	44 032	AC	60 416	EC
11 520	2D	27 904	6D	44 288	AD	60 672	ED
11 776	2E	28 160	6E	44 544	AE	60 928	EE
12 032	2F	28 416	6F	44 800	AF	61 184	EF
12 288	30	28 672	70	45 056	B0	61 440	F0
12 544	31	28 928	71	45 312	B1	61 696	F1
12 800	32	29 184	72	45 568	B2	61 952	F2
13 056	33	29 440	73	45 824	B3	62 208	F3
13 312	34	29 696	74	46 080	B4	62 464	F4
13 568	35	29 952	75	46 336	B5	62 720	F5
13 824	36	30 208	76	46 592	B6	62 976	F6
14 080	37	30 464	77	46 848	B7	63 232	F7
14 336	38	30 720	78	47 104	B8	63 488	F8
14 592	39	30 976	79	47 360	B9	63 744	F9
14 848	3A	31 232	7A	47 616	BA	64 000	FA
15 104	3B	31 488	7B	47 872	BB	64 256	FB
15 360	3C	31 744	7C	48 128	BC	64 512	FC
15 616	3D	32 000	7D	48 384	BD	64 768	FD
15 872	3E	32 256	7E	48 640	BE	65 024	FE
16 128	3F	32 512	7F	48 896	BF	65 280	FF

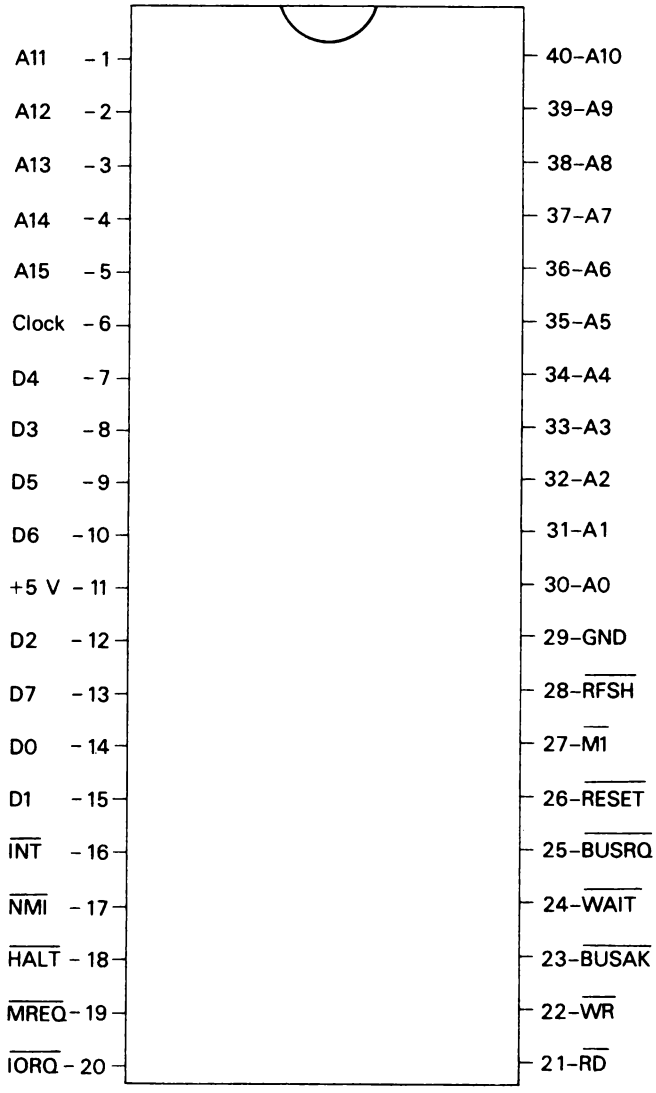
Appendice

I segnali dello Z80

C

A0–A15: linee indirizzi (vedi Capitoli 1 e 2).
D0–D7: linee dati (vedi Capitoli 1 e 2).
INT: la linea di interrupt mascherabile, che scandisce la tastiera ogni cinquantesimo di secondo durante l'esecuzione di programmi in codice-macchina.
NMI: la linea di interrupt non mascherabile, che interrompe l'esecuzione del programma in codice-macchina in corso e lancia l'esecuzione di opportune routine di gestione dell'interrupt.
HALT: questa linea è attiva durante l'esecuzione di routine in codice-macchina.
MREQ: questa linea viene attivata ogni volta che vi è scambio di dati fra il processore e la memoria.
IORQ: linea di richiesta di input/output, è attivata da istruzioni di ingresso o uscita (IN, OUT, LOAD, SAVE...)
RD: la linea di lettura è attiva quando è in corso un'operazione di lettura dalla memoria.
WR: la linea di scrittura è attiva quando è in corso un'operazione di scrittura in memoria.
BUSAK: la linea di cessione del bus lavora insieme a BUSRQ (linea di richiesta del bus) per soddisfare eventuali richieste dell'utente.
WAIT: la linea di attesa rallenta le operazioni per permettere l'uso di memorie lente.
RESET: la linea di reset, all'accensione o quando necessario, inizializza lo Z80 ad uno stato predeterminato.
M1: la linea di fetch è attiva durante l'acquisizione dalla memoria di un byte di istruzioni o di dati.

RFSH: la linea di refresh controlla il "rinfresco" dei dati nelle memorie dinamiche e viene usata per generare i segnali video.
CLOCK: la frequenza di clock fornita dallo Spectrum è di 3.5 MHz, cioè 7 impulsi ogni 2 milionesimi di secondo.
Alimentazione: lo Z80 richiede un'alimentazione a +5 V fra il piedino 11 il piedino 29 (massa).

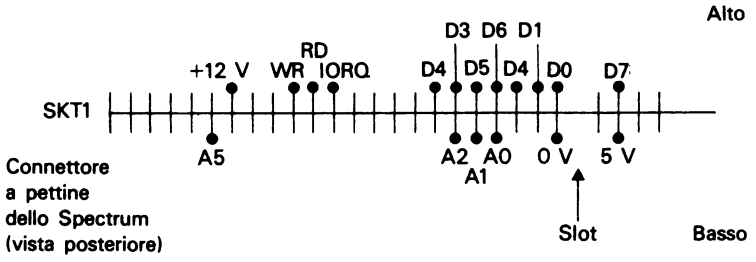


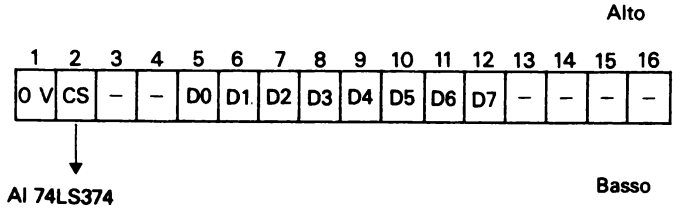
AO-A15 sono i 16 piedini del bus indirizzi
DO-D7 gli 8 dati

Appendice

Pin del connettore a pettine per i tre progetti principali

D





DAC SKT2

1	2	3	4	5	6	7	8	9	10	11	12
0 V	–	+10	+5	D7	D6	D5	D4	D3	D2	D1	D0

Latch SKT3

Appendice

Dati sui componenti

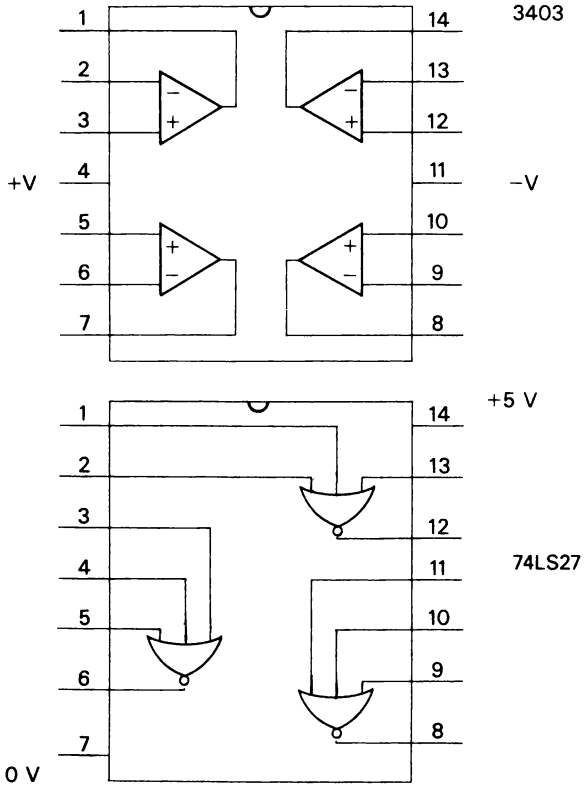


Figura E.1 (continua).

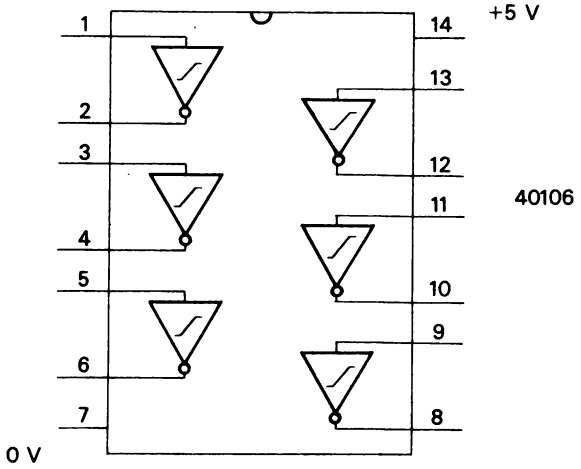


Figura E.1

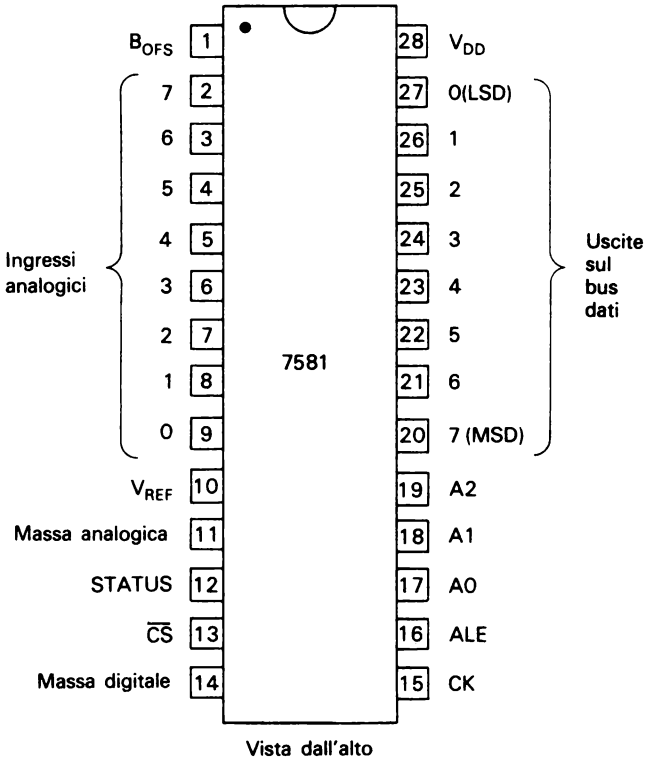
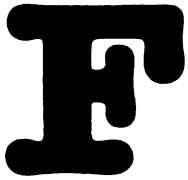


Figura E.2

Appendice

Elenco dei componenti



CAPITOLO 3

ADC	1	Basetta stampata tipo RS433-955
	1	Connettore a pettine da wire-wrap da 2×43 poli
	8	Connettori da 2 mm per circuito stampato (usati per punti di controllo)
	8	Spine da 2 mm
	2	Zoccoli per circuiti integrati a 14 piedini
	1	Zoccolo per circuiti integrati a 28 piedini
	1	Circuito integrato ADC 7581
	1	Circuito integrato 74LS27 triplo NOR a tre ingressi
	1	Circuito integrato 40106 hex Schmitt trigger inverter
	4	Diodi 1N4148
	1	Diodo Zener da 10 V
	3	Condensatori ceramici da 33 nF
	1	Condensatore ceramico da 68 pF
	1	Condensatore da 10 F 25 V
	1	Resistenza da 680 1/8 W
	1	Resistenza da 22 1/8 W
		Filo sottile per connessioni isolato in PVC
		Filo sottile per connessioni nudo
Opzionale		1 Connettore a pettine a 12 o 16 poli con terminali a saldare (per la basetta DAC).

CAPITOLO 4

DAC	1	Basetta stampata RS433-955
	1	Connettore a pettine da wire-wrap a 12 poli

	5	Connettori per circuito stampato da 2 mm
	5	Spinotti da 2 mm
	2	Zoccoli per circuiti integrati a 14 piedini
	1	Zoccolo per circuiti integrati a 16 piedini
	1	Zoccolo per circuiti integrati a 20 piedini
	1	Circuito integrato DAC0801
	1	Circuito integrato 74LS374 latch
	1	Circuito integrato 74LS04 hex inverter
	1	Circuito integrato 3403 quadruplo amplificatore operazionale
	1	Resistenza da 470 k 1/8 W
	4	Resistenze da 10 k 1/8 W
	1	Resistenza da 4.7 k 1/8 W
	1	Resistenza da 3.9 k 1/8 W
	1	Condensatore ceramico da 0.1 μF
		Filo sottile per connessioni isolato in PVC
		Filo sottile per connessioni nudo
Alimentatore	1	Trasformatore di rete 240 V/6 V+6 V 6VA
		Interruttore e fusibile in contenitore di sicurezza
	2	Ponti rettificatori 6A miniatura al silicio
	2	Condensatori elettrolitici 1000 μF 25V
	1	Regolatore 7805
	1	Regolatore 7985
	2	Condensatori elettrolitici 10 μF 25V
Latch	1	Basetta stampata RS433-955
	1	Connettore a pettine da wire-wrap 2×43 poli
oppure	8	Connettori per circuito stampato da 2 mm
	1	Connettore a pettine a 12 poli per collegare gli optoisolatori
	1	Zoccolo per circuiti integrati a 20 piedini
	1	Zoccolo per circuiti integrati a 14 piedini
	1	Circuito integrato 74LS374 latch
	1	Circuito integrato 74LS27 triplo NOR a tre ingressi
		Filo sottile per connessioni isolato in PVC.
		Filo sottile per connessioni nudo.

Appendice

Adattamenti per lo Spectrum 16 K

	64 K	16 K	(pagina)
CAPITOLO 2			
	40000	32000	
	(58327) 17	(25088)17	**30**
	215	0	
	227	98	
	33	33	
	215	0	
	227	98	
CAPITOLO 3			
penna ottica	10 CLEAR 44500	25000	**56**
	40 65240	32000	
	60 DATA 17,215,227...	17,0,98	
	70 DATA...33,215,227...	33,0,98	
	90 LET l=USR 65240	32000	
	210 LET l=USR 65261	32021	
penna ottica in codice-macchina: come penna ottica, tranne			
	100 LET l=USR 65240	32000	**61**
	120 LET l=USR 65282	32042	

	64 K	16 K	(pagina)
	140 LET l=USR 65261	32021	
	ld 63744,a 50	50	
	0	0	
	249	124	
	80 DATA...		
	122,50,0,249,123,50,1,	122,50,0,124,123,	
	249,201	50,1,124,201	
	150....63745....63744	....31745....31744....	
oscillo-scopio	ld de,40960	26112	**62**
	50 DATA 17, 0, 160...	17,0,102....	
	30 POKE (64000+x),n	32000	
	70 LET l=USR 64000	32000	
	100 PEEK (40960+x)	26112	

CAPITOLO 4

sequencer e timer

	POKE or PEEK		
	40000 and 40001	32000 and 32001	**80,83,84**

CAPITOLO 5

registra-tore	ld de, 40960	25088		**93,95**
	50and 55 DATA 17,0,160	17,0,98...		
	30 POKE(64000+x),n	32000		
	70 LET l=USR 64000	32000	+	
	90 LET l=USR 64029	32029		

Spectrum parlante — non trasportabile sul 16 K

robot	1 ld de,40960	25088		**102**
	50 DATA 17,0,160	17,0,98		
	30 POKE(64000+x),n	32000	+	

	64 K	16 K	(pagina)
	70 POKE 64002,	32002	
	90, 160 LET l=USR 64000	32000	
	110 40960	25088	
	150 POKE 64002	32002	
	180 PEEK 43008	27136	
robot in codice-macchina			
	POKE 64000	32000	**108**
robot 2: nessuna variazione			
eco	ld de,40960	25088	**110,111**
	50 DATA 17,0,160...	17,0,98...	
	...195,255,249	195,255,124	
	30 POKE 64000	32000	
	80 LET l=USR 64000	32000	
treno	POKE 65240	32512	**113**
	6 DATA 17,215,227...	17,0,98...	
	...33,215,227...	...33,0,98...	
	4 POKE 65240	32512	
	9 LET l=USR 65240	32512	
	190 POKE 40000	32000	
	40001	32001	
	305, 380 LET l=USR 65261	32533	
	310 PEEK 40000	32000	

riscaldamento: come sequencer

antifurto: nessuna variazione

Appendice

Basette a circuito stampato

H

Presentiamo i disegni dei circuiti stampati delle basette ADC, DAC e latch.
Le basette sono a doppia faccia e sono reperibili presso ogni rivenditore di materiale elettronico, oppure da Maplin Electronic Supplies Ltd, PO Box 3, Rayleigh, Essex SS6 8LR, Gran Bretagna.
Sono disponibili scatole di montaggio complete di tutti i componenti.
Per maggiori dettagli scrivete alla Maplin Electronic Supplies che effettua anche vendite per corrispondenza.

CONVERTITORE ANALOGICO-DIGITALE (ADC)

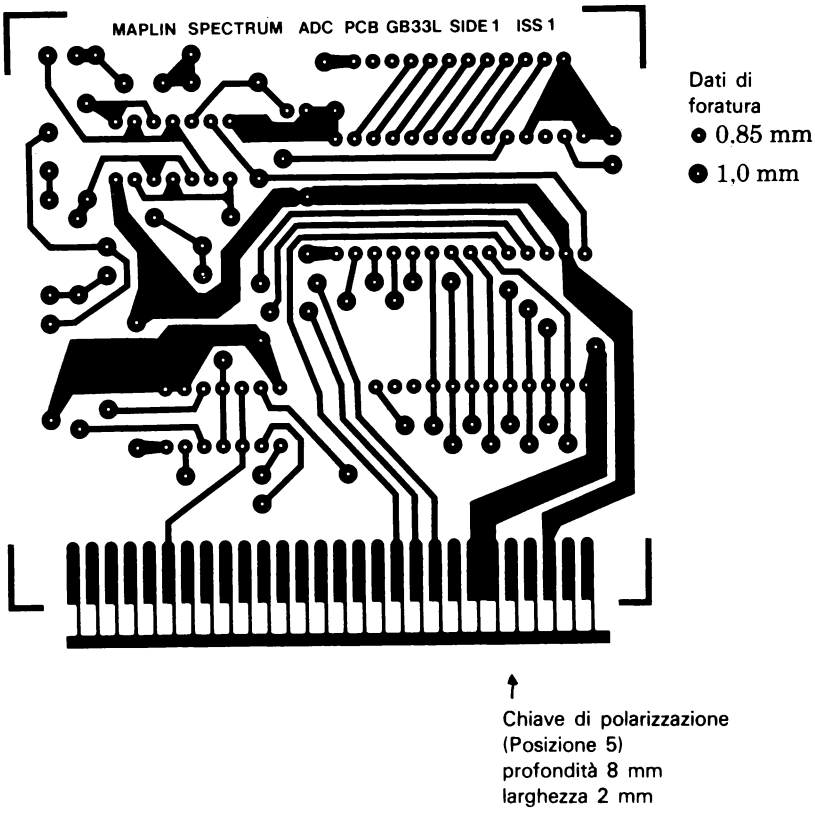
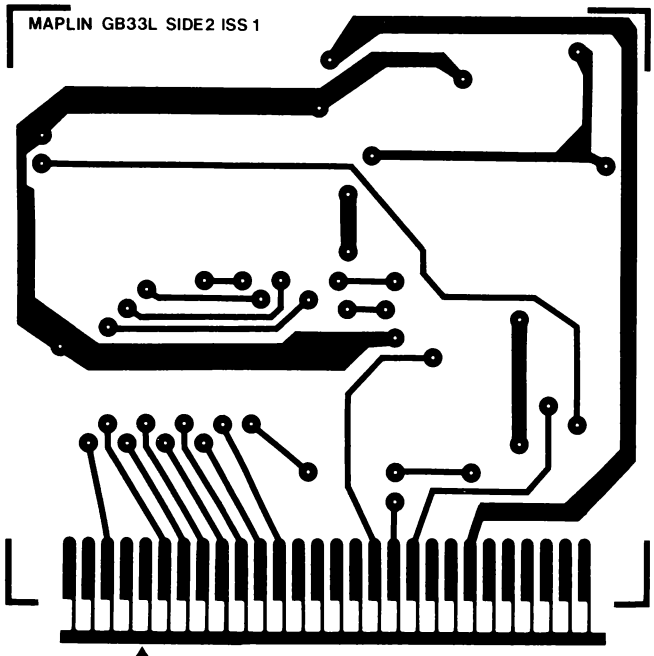


Figura H.1 Il circuito stampato, grandezza naturale, lato 1 e lato 2 (pagina 163).

(continua)



Chiave di polarizzazione
(Posizione 5) profondità 8 mm larghezza 2 mm

Figura H.1

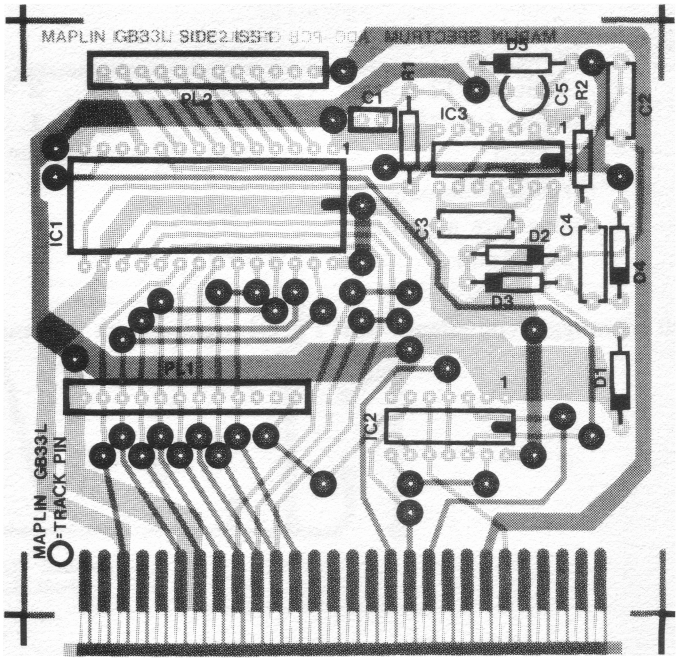
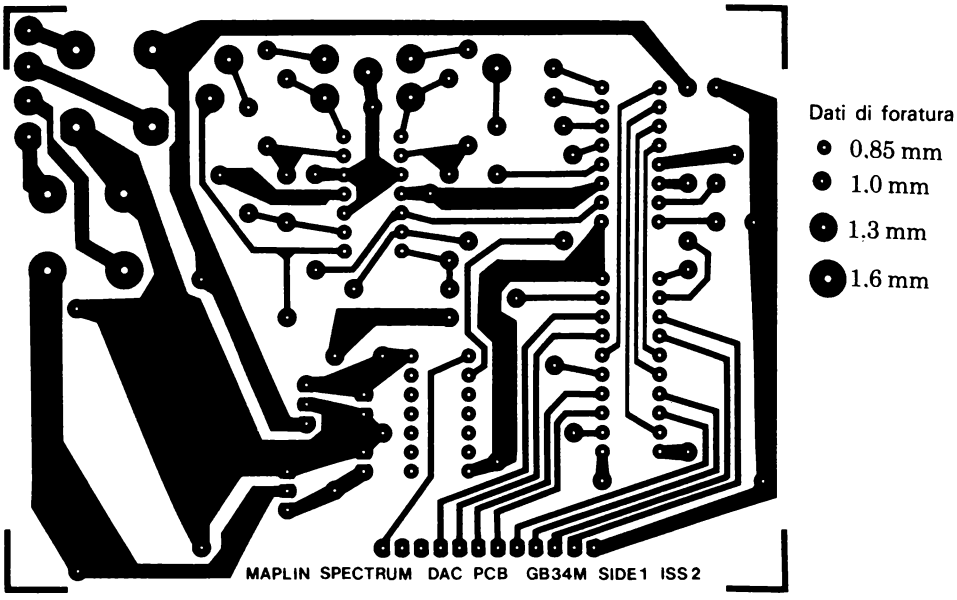


Figura H.2 Disposizione dei componenti, vista dal lato 2.

	Caratteristiche	Numero di catalogo Maplin	Quantità
Resistenze	0.4 W 1% a film metallico		
R1	680 R	(M680R)	1
R2	22k	(M22K)	1
Condensatori			
C1	68 pF Ceramico	(WX54J)	1
C2,3,4	33 nF Poliestere	(BX73Q)	3
C5	10 μF 16V Tantalio	(WW68Y)	1
Semiconduttori			
IC1	7581ADC	(QY56L)	1
IC2	72LS27	(YF18U)	1
IC3	40106BE	(QW64U)	1
D1,2,3,4	1N4148	(QL80B)	4
D5	BZY88C10V	(QH14Q)	1
Varie			
PL1,2	Connettore Mini con 12 poli	(YW14Q)	2
	Zoccolo DIL 28 pin	(BL21X)	1
	Pin sfusi	(FL82D)	42 pin (50 per confezione)
	Circuito stampato ADC	(GB33L)	1
	Connettore a pettine 2×28 poli	(YG23A)	1

La scatola di montaggio contenente tutti i pezzi necessari ha il numero di codice LK26D.

CONVERTITORE DIGITALE-ANALOGICO (DAC)



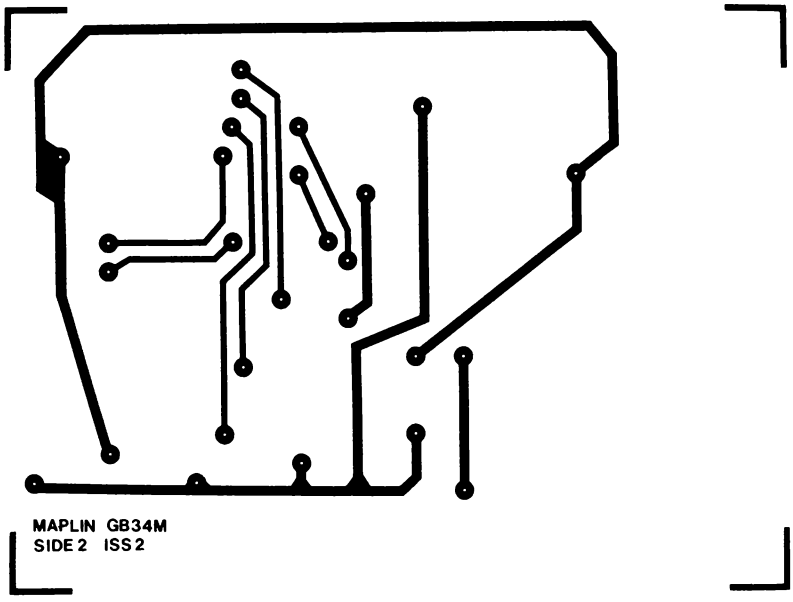


Figura H.3 Circuito stampato, grandezza naturale, lato 1 e lato 2.

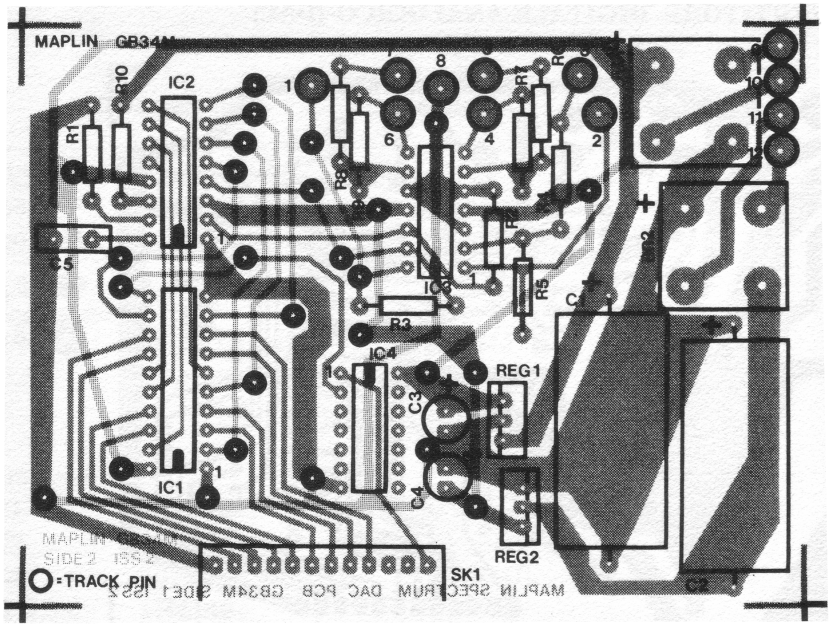
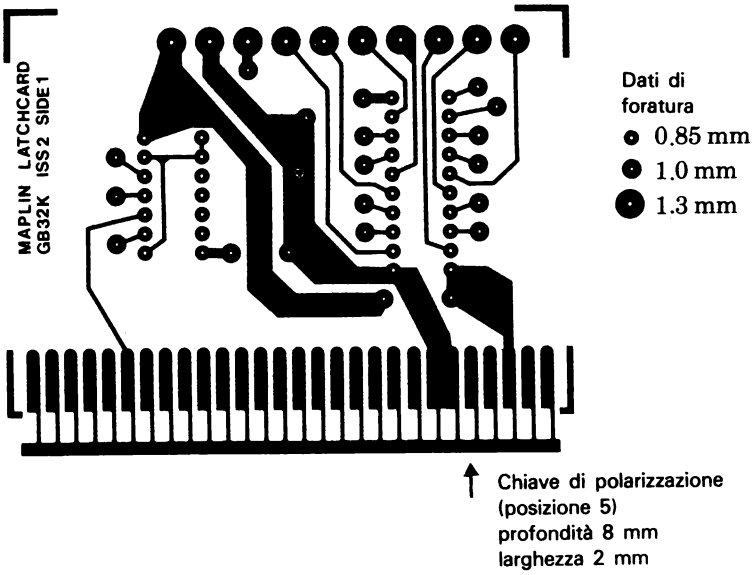


Figura H.4 Disposizione dei componenti del DAC, vista dal lato 2.

	Caratteristiche	Numero di catalogo Maplin	Quantità
Resistenza	0,4 W 1% a film metallico		
R1	4k7	(M4K7)	1
R2,3,4,6,8	10k	(M10K)	5
R5,7,9	470k	(M470K)	3
R10	3k9	(M3K9)	1
Condensatori			
C1,2	1000 μF 25 V elettrolitico assiale	(FB83E)	2
C3,4	10 μF 35 V PC Elettrolitico	(FF04E)	2
C5	22 nF Ceramico	(WX78K)	1
Semiconduttori			
IC1	74LS374	(YH16S)	1
IC2	DAC0801	(QQ01B)	1
IC3	3403	(QH51F)	1
IC4	74LS04	(YF04E)	1
REG 1	μA7805UC	(QL31J)	1
REG 2	μA7905UC	(WQ92A)	1
BR1, 2	PW01	(WQ57M)	2
Varie			
SK1	Connettore Minicon 12 poli	(YW30H)	1
	Veropin 2140	(FL20W)	12 pin (100 per confezione)
	Pin sfusi	(FL82D)	27 pin (50 per confezione)
	Circuito stampato DAC	(GB34M)	1
	Zoccoli DIL 16 pin	(BL19V)	1
T1	Trasformatore 6 V	(WB06G)	1
	Filo di alimentazione	(BL00A)	1 confezione (si può usare da BL00A a BL10L)

La scatola di montaggio contenente tutti i pezzi necessari ha il numero di codice LK25C

SCHEDA LATCH



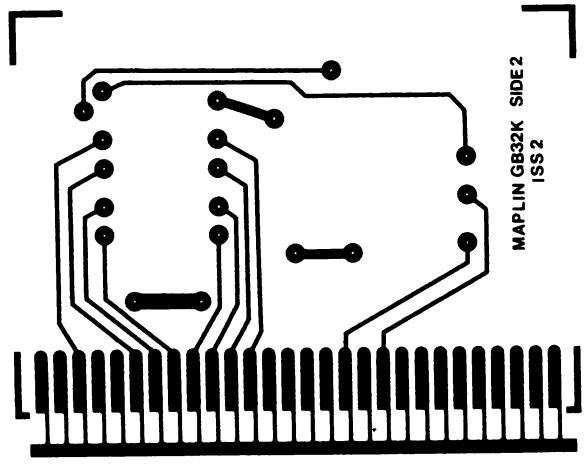


Figura H.5 Circuito stampato, grandezza naturale, lato 1 e lato 2.

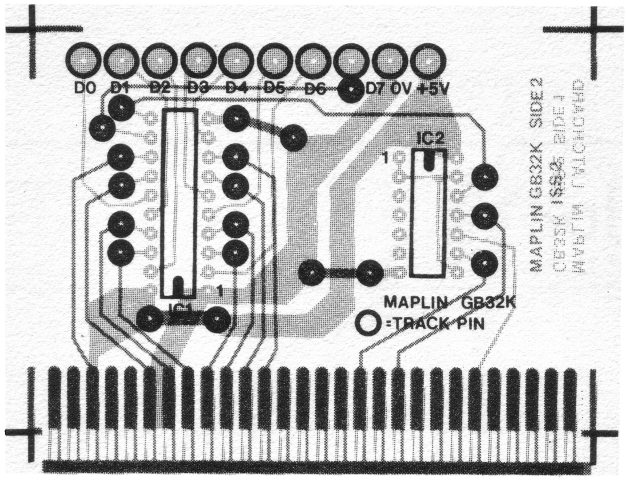


Figura H.6 Disposizione dei componenti, vista dal lato 2.

	Caratteristiche	Numero di catalogo Maplin	Quantità
Semiconduttori			
IC1	74LS374	(YH16S)	1
IC2	74LS27	(YF18U)	1
Varie			
	Scheda latch PCB	(GB32K)	1
	Veropin 2141	(FL21X)	10 pin (100 per confezione)
	Pin sfusi	(FL82D)	20 pin (50 per confezione)
	Connettore a pettine 2×28 poli	(YG23A)	1

La scatola di montaggio contenente tutti i pezzi necessari ha il numero di codice LK24B.

Software

Cassette disponibili:

G. Bishop, **Spectrum Interfacing Programs**

ISBN 07-084709-6

Questa cassetta contiene tutti i programmi, in versione originale inglese, presentati in questo libro e fa risparmiare ore e ore di lavoro sulla tastiera senza rischio di dover correggere più volte i listati. Utile in particolar modo per le lunghe routine in codice-macchina.

ZX Spectrum Machine Code Assembler

ISBN 88-7700-901-2

Un completo programma assemblatore indispensabile per compilare i propri programmi scritti in Assembler secondo le specifiche descritte nel libro di T. Woods, *L'Assembler per lo ZX Spectrum.* È il naturale complemento del libro e permette di ottenere dei programmi molto più veloci e compatti dei corrispondenti in BASIC.
La cassetta è corredata da un manuale d'uso in italiano.

Nella stessa serie:

— C.A. Street, *La gestione delle informazioni con lo ZX Spectrum*

— T. Woods, *L'Assembler per lo ZX Spectrum*

— J. Heilborn e R. Talbott, *Guida al Commodore 64*

— R. Jeffries, G. Fisher e B. Sawyer, *Divertirsi giocando con il Commodore 64*

— H. Mullish e D. Kruger, *Il BASIC Applesoft*

— H. Peckham, *Il BASIC e il PC-IBM in pratica*

— H. Peckham, *Il BASIC e il Commodore 64 in pratica*

Di prossima pubblicazione:

— S. Nicholls, *Giochi in Assembler con lo ZX Spectrum*

— S. Skier, *L'Assembler per il Vic 20 e il Commodore 64*

— S. Kamins e M. Waite, *Programmate meglio il vostro Apple*

— M. Williams, *Inventa i tuoi giochi con lo ZX Spectrum*

L'interfacciamento del personal computer ai più diversi strumenti e dispositivi è un'applicazione poco diffusa.
Progetti hardware con lo ZX Spectrum è stato concepito proprio per aprire agli appassionati le porte del mondo del controllo automatico dei sistemi, in maniera facilmente comprensibile e immediatamente applicabile.
Questo volume rappresenta infatti molto di più di un testo sulle possibilità di interfacciamento dello Spectrum; vi sono descritti in dettaglio un convertitore analogico-digitale e uno digitale-analogico che possono essere collegati alla porta di espansione dello ZX Spectrum. Con questi è possibile creare esposimetri e penne ottiche, termometri di precisione e antifurti, joystick e simulatori di voce, oppure guidare il braccio meccanico di un robot o un trenino elettrico.
Il libro contiene tutti gli schemi elettronici e l'indicazione dei componenti necessari per la realizzazione dei progetti descritti, nonché i listati dei programmi di gestione delle schede da collegare alla porta di espansione.

McGraw Hill

Lire 17 000
(IVA 2% inclusa)

ISBN 88 7700 005 8

G. Bishop **Progetti hardware con lo ZX SPECTRUM**

McGraw-Hill